Auto prompting without training labels: An LLM cascade for product quality assessment in e-commerce catalogs

Soham Satyadharma[1], Fatemeh Sheikholeslami [1], Swati Kaul[1], Aziz Umit Batur[2], Suleiman A. Khan[1]

[1] Amazon Catalog AI, [2] formerly at Amazon Catalog AI

ssatyadh@amazon.com, shfateme@amazon.com, kauswati@amazon.com, umitbatur@gmail.com, suleimkh@amazon.com

Abstract

We introduce a novel, training free cascade for auto-prompting Large Language Models (LLMs) to assess product quality in e-commerce. Our system requires no training labels or model fine-tuning, instead automatically generating and refining prompts for evaluating attribute quality across tens of thousands of product category–attribute pairs. Starting from a seed of human-crafted prompts, the cascade progressively optimizes instructions to meet catalog-specific requirements. This approach bridges the gap between general language understanding and domain-specific knowledge at scale in complex industrial catalogs. Our extensive empirical evaluations shows the auto-prompt cascade improves precision and recall by 8–10% over traditional chain-of-thought prompting. Notably, it achieves these gains while reducing domain expert effort from 5.1 hours to 3 minutes per attribute - a 99% reduction. Additionally, the cascade generalizes effectively across five languages and multiple quality assessment tasks, consistently maintaining performance gains.

1 Introduction

Product catalogs are the cornerstone of e-commerce, where the accuracy of product information directly impacts user experience and business outcomes (Amsl et al., 2023; Lv and Liu, 2022; Sadinle et al., 2022). Each product is defined by unstructured attributes (free-text like titles or descriptions) and structured attributes (feature-value pairs like color or size) (Nikolakopoulos et al., 2023). A fundamental challenge in maintaining catalog quality is ensuring the alignment between these two attribute types, as inconsistencies frequently arise from discrepancies between seller descriptions and how attributes are formally modeled (Schmidts et al., 2020). This complexity is twofold: First, inherent semantic ambiguities make verification difficult. For example, identifying the base material of a walking stick from the description, *wood construction with a steel spike and rubber tip*, requires disambiguating material of each component. Second, attribute values often rely on implicit sequential reasoning, such as inferring that a product talks about pet food or human food, and using this inference to predict if valid age is *puppies* or *young adult*. For additional examples see: §C.4.

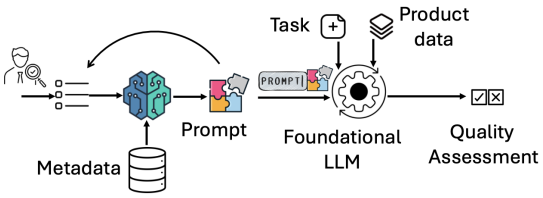

Figure 1: Auto-prompt LLM cascade for quality measurement. The cascade takes metadata and manually created few-shot examples to iteratively generate prompts, subsequently used to classify the attribute value.

These alignments and nuances vary for each product category-structured attribute (PC-SA) pair, which exhibits distinct characteristics and semantic interpretations. Here PC is a group of similar products (e.g., speakers or shirts), and SA is a structured attribute (e.g., color or material). While Large Language Models (LLMs) offer strong reasoning capabilities (Zubiaga, 2024; Hadi et al., 2023; Min et al., 2023; Huang et al., 2024; Li et al., 2023), steering them for such a specialized task with ten's of thousands of implicit nuances remains a complex challenge. This is further compounded by misalignment of LLM's general knowledge with specialized terminology and quality expectations of e-commerce stores. Single, general-purpose solutions (zero-shot or few-shot) struggle to effectively capture these variations (Jiang et al., 2024b). A promising direction is to steer the LLM to determine correct values through domain-aware, case-specific instructions for each PC-SA pair. This is a manual task suitable for subject matter experts and estimated to require over 3,000 human-days for over 12,000 PC-SA pairs.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 937–953
November 4-9, 2025

Previous attempts to address these challenges have shown promise but faced limitations. For instance, MetaBridge (Wang et al., 2020) combines meta-learning with latent variable modeling to verify attributes but requires an exceedingly large number of labeled training samples for each PC-SA, rendering it practically infeasible at the scale of tens of thousands of PC-SA. On the other hand, zero-shot Chain-of-Thought (CoT) approaches have helped in simpler tasks (Wei et al., 2022; Kojima et al., 2022; Zhang et al., 2022b; Wang et al., 2022), but they lack the granularity to handle the thousands of heterogeneous relationships between PCs and SAs, which are needed to ensure domain-specific terminology, contextual nuances, and the hierarchical nature of product taxonomies.

To address these challenges, we introduce an innovative LLM cascade for large scale product quality assessment without any training labels or model fine-tuning (Fig. 1). Our approach steers off-the-shelf LLMs by iteratively generating and refining tens of thousands of prompts for two catalog quality tasks: correctness and applicability (§4.1). A core component of the prompts are tailored PC-SA instructions which steer the LLM in making product quality assessments. Bootstrapping from a small set of human-authored examples, the cascade creates instructions for each SA across multiple PCs (e.g., material-sofas, material-screws). The cascade iteratively refines these instructions by incorporating detailed metadata, such as PC and SA definitions, valid-values etc. In the remainder of the paper, we use the terms prompt and instruction interchangeably.

The contributions of this paper are three fold. (1) We introduce a novel, training-free LLM cascade that automatically generates and refines prompts for e-commerce quality assessment, requiring no labeled data or model fine-tuning. (2) We demonstrate that our approach improves precision and recall by 8–10% over the baseline across tens of thousands of diverse PC-SA pairs, while reducing human prompting effort by 99%. (3) We establish that our method generalizes effectively across multiple quality tasks, foundational LLMs, and languages, all without the need for task or language specific training labels.

2 Related work

Prompt engineering: In-context learning through hand-crafted prompts have been shown to improve LLM performance. Approaches like Chain-of-Thought prompting (Wei et al., 2022; Ma et al., 2023) and few-shot learning (Brown et al., 2020; Radford et al., 2019) enhance LLM reasoning, though they rely on manual engineering and offer limiting scalability to write thousands of prompts. Our work builds on these by automating prompt generation at scale to create tens of thousands of domain-specific prompts while retaining benefits of structured prompt design offered by both.

Automated prompt generation: Automated prompt generation has shown promise in improving prompts, however, most methods focus on general-purpose tasks (Zhou et al., 2022; Opsahl-Ong et al., 2024) rather than domain-specific improvements. Approaches like reinforcement learning (Deng et al., 2022) and evolutionary algorithms (Guo et al., 2023) optimize prompts based on feedback; while clustering and failure-driven rules help select optimal prompts from synthetic candidates (Do et al., 2024; Gao et al., 2025). PRISM (He et al., 2024) and PromptGen (Zhang et al., 2022a) use iterative approaches for in-context learning. However, these techniques require labeled development sets or error feedback for optimization, whereas our method generates domain-specific prompts directly from knowledge hierarchies without such supervision.

Recently, hybrid approaches combining prompt engineering and fine-tuning have shown promising results, for example integrating prompt tuning with Bayesian regression (Wang et al., 2025), fine-tuning (Soylu et al., 2024) and reinforcement learning (Byun et al., 2024; Kong et al., 2024). However, unlike ours, these approaches require training labels for optimizing both models and prompts which is prohibitive at the scale of tens of thousands of PC-SA's.

LLMs for e-commerce: LLMs have shown benefit in several e-commerce applications such as, improving recommendation systems (Maragheh et al., 2023), knowledge graph completion (Chen et al., 2023), search (Rokon et al., 2024), product discovery (Wang et al., 2024), product matching (Herrero-Vidal et al., 2024), categorization (Cheng et al., 2024; Kathiriya et al., 2023), and attribute extraction (Baumann et al., 2024). However, no prior work has investigated using LLMs for assessing product quality.

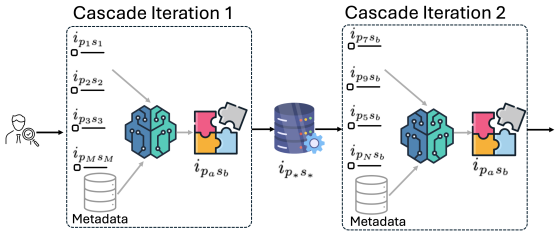

Figure 2: Example of auto prompting cascade for instruction generation of PC denoted as p_a and attribute as s_b across two iterations. Iteration 1 utilizes M manually created global PC-SA instructions and is repeated for M PC definitions to generate M PC-SA instructions for the s_b attribute that are used by iteration 2 to combine these examples with the definition for p_a and s_b to produce the PC-SA instruction for p_a - s_b.

3 Method

3.1 Problem formulation

Let $\mathcal{A}$ be the universe of products, where each product $a \in \mathcal{A}$ is characterized by a tuple $(p_a, \{s_{a_i}\}_{i=1}^{|S|}, u_a)$, where $p_a \in \mathcal{P}$ is the PC, $s_{a_i} \in \mathcal{S}$ denotes the i-th structured attribute in the set of structured attributes (SAs) for product a, and $u_a \in \mathcal{U}$ denotes the unstructured attributes (UAs) for product a. By dropping subscript a for simplicity, we define the task of quality assessment of SA s_i for a given product a as learning the classification function

$$f : \mathcal{P} \times \mathcal{S} \times \mathcal{U} \rightarrow \mathcal{R} \times \mathcal{C} \tag{1}$$

where $\mathcal{C}$ represents the classification decision and $\mathcal{R}$ represents the LLM's rationale for the decision.

This function is implemented through a text-to-text transformation, where prompts are crafted for a given (p, s, u) using LLMs. We design the prompts to have placeholders for input data and PC-SA specific instructions, therefore reducing the auto-prompting task to optimizing the *instructions* to the quality assessment LLM. To reduce the search space and parameterize the problem, we structure the instructions at PC-SA level, considering instructions to be dependent only on the SA s_i and PC p given by product a, and model it as independent from the UA value u. Note that UA values- titles, product descriptions, and images - are mainly what the correctness of SA s gets assessed against, hence the classifier f is a function of u, however the *structure* of the prompt is modeled independently.

Thus, given the PC and SA of interest of the given product, instruction creation is modeled by $\pi : \mathcal{P} \times \mathcal{S} \rightarrow \mathcal{I}$ where $\mathcal{P}$ and $\mathcal{S}$ are the set of all PCs and SAs respectively, and $\mathcal{I}$ represents the instruction space. These instructions are then fed into the text-to-text transformation function $T : \mathcal{I} \times \mathcal{U} \rightarrow \mathcal{Y}$, to create an element of the output token space $\mathcal{Y}$. The end-to-end pipeline can now be expressed as

$$f(p, s, u) = g(T(\pi(p, s), u)); \quad g : \mathcal{Y} \rightarrow \{\mathcal{R}, \mathcal{C}\}.$$

3.2 CoT prompting approach

We adapt a Chain-of-Thought (CoT) prompting approach (Wei et al., 2022; Ma et al., 2023), assuming that an LLM's inherent contextual understanding from pre-training can resolve ambiguity in our task. This simplifies our general prompt function from Equation 1 to $f(p, s, u) = g(T(p, s, u))$. Here, we rely on the model's implicit knowledge of the PC-SA pair (p, s) to produce a high-quality transformation $T(p, s, u)$ without explicit instructions. Our zero-shot prompt incorporates intermediate reasoning steps to guide the classification, which further helps the enhanced subsequent techniques detailed next.

3.3 Human engineered PC-SA instructions

To capture nuanced requirements and store-specific expectations at a PC-SA level, we manually engineer a set of "golden" instructions with the help of domain experts. These instructions cover 2,388 PC-SAs, which we select based on PC frequency, and serve as our implementation of the mapping function $\pi(p, s)$ defined in §3.1. To manage the significant manual effort, we adopt a hierarchical approach, authoring instructions at the more general SA level rather than for each PC-SA pair.

Each instruction provides comprehensive guidance to the LLM by including three key components: (1) a formal definition of the SA's semantic scope within e-commerce; (2) clear guidelines for handling edge cases and ambiguity; and (3) domain-specific constraints that may differ from the LLM's general pre-trained knowledge.

3.4 Cascaded PC-SA instruction generation

Manually creating instructions for each PC-SA pair improves LLM comprehension but is unscalable for large catalogs that contain tens of thousands of such pairs. To address this challenge, we introduce an auto-prompt cascade for instruction generation by leveraging a minimal set of manually crafted instructions to efficiently generate nuanced, PC-SA specific instructions at scale.

Algorithm 1 Cascaded few shot prompt generation

$\mathcal{I}_{\text{H}} \leftarrow$ set of hand crafted instructions
for $s \in \mathcal{S}$ **do** ▷ iteration 1
  $\mathcal{I}_{A^{(1)}} \leftarrow \emptyset$ ▷ Initialize $\mathcal{I}_{A^{(1)}}$ as the set of auto-generated instructions for attribute s
  Randomly sample M product categories from $\mathcal{P}$ to get $\mathcal{P}' = \{p_1, ..., p_M\}$
  for $p_x \in \mathcal{P}'$ **do** ▷ Use LLMs to generate automated instructions for (p_x, s)
    $\mathcal{I}_{A^{(1)}} \leftarrow \mathcal{I}_{A^{(1)}} \cup \text{LLM}(d(p_x), d(s), \mathcal{I}_{\text{H}}^{M})$
  end for
end for
for $\tau \in \{2, ..., T\}$ **do** ▷ iteration τ
  $\mathcal{I}_{A^{(\tau)}} \leftarrow \emptyset$ ▷ Initialize the set of final PC-SA instructions
  for $s \in \mathcal{S}$ **do** ▷ Use $\mathcal{I}_{A^{(\tau-1)}}$ as few shot examples to generate instructions for all (p, s) pairs
    for $p \in \mathcal{P}$ **do**
      $\mathcal{I}_{A^{(\tau)}} \leftarrow \mathcal{I}_{A^{(\tau)}} \cup \text{LLM}(d(p), d(s), \mathcal{I}_{A^{(\tau-1)}}^{M})$
    end for
  end for
end for
return $\mathcal{I}_{A^{(T)}}$

Let us formally define the instruction generation problem. Let d be the definition function, such that $d_{\mathcal{P}}(p) : \mathcal{P} \rightarrow \mathcal{D}_{\mathcal{P}}$ denotes the definition of the PC p, and $d_{\mathcal{S}}(s) : \mathcal{S} \rightarrow \mathcal{D}_{\mathcal{S}}$ denotes the definition of the SA s. We will drop $\mathcal{S}$ and $\mathcal{P}$ subscripts to simplify notation. Let $\mathcal{I}_{\text{H}} \subset \mathcal{I}$ denote the space of human-crafted PC-SA instructions and $\mathcal{I}_{\text{A}} \subset \mathcal{I}$ denote the space of automatically generated PC-SA instructions, where $\mathcal{I}$ is the space of all instructions.

In order to generate an instruction for a given PC-SA (p_a, s_b) and capture the nuances for SA s_b specific to product type p_a, we propose to generate an automated PC-SA instruction $i'_{p_a s_b}$ by utilizing K few-shot examples over K different product categories for the same SA s_b, denoted by $i_{p_k s_b} \in \mathcal{I}_{\text{H}}, 1 \leq k \leq K$. This is tantamount to modeling the instruction generation function as

$$\pi'(d(p_a), d(s_b), i_{p_1 s_b}, \cdots, i_{p_K s_b}) = i'_{p_a s_b}, \quad \forall a \in \{1, \cdots, |\mathcal{P}|\}, b \in \{1, \cdots, |\mathcal{S}|\} \quad (2)$$

This mapping $\pi' : \mathcal{D}_{\mathcal{P}} \times \mathcal{D}_{\mathcal{S}} \times \mathcal{I}_{\text{H}}^{K} \rightarrow \mathcal{I}_{\text{A}}$ is our proposed efficient proxy of the more ambiguous instruction mapping function $\pi : \mathcal{P} \times \mathcal{S} \rightarrow \mathcal{I}$. However, this approach is intractable in practice as it necessitates K instructions for each SA, leading to a total of $\mathcal{O}(K \times |\mathcal{S}|)$ hand-crafted instructions.

To address this limitation, we propose a cascaded instruction generation framework where we utilize an LLM to assist the above using a much smaller set of M hand-crafted instructions. Fig 2 illustrates an example through the first two iterations of the cascade. In iteration 1, we leverage M manually created PC-SA instructions to generate M few shot examples per SA through a mapping function $h_1 : \mathcal{D}_{\mathcal{P}} \times \mathcal{D}_{\mathcal{S}} \times \mathcal{I}_{\text{H}}^{M} \rightarrow \mathcal{I}_{\text{A}}$ defined as

$$h_1(d(p_x), d(s_b), i_{p_1 s_1}, \cdots, i_{p_M s_M}) = i'_{p_x s_b}, \quad x \in \{1, \cdots, M\}, b \in \{1, \cdots, |\mathcal{S}|\} \quad (3)$$

In iteration 2, we utilize M automatically generated instructions as few shot examples at an SA level for the SA s_b to produce PC-SA instructions for target pair (p_a, s_b) through mapping function $h_2 : \mathcal{D}_{\mathcal{P}} \times \mathcal{D}_{\mathcal{S}} \times \mathcal{I}_{\text{A}}^{\text{M}} \rightarrow \mathcal{I}_{\text{A}}$ defined as

$$h_2(d(p_a), d(s_b), i'_{p_1 s_b}, \cdots, i'_{p_M s_b}) = i'_{p_a s_b}, \quad a \in \{1, \cdots, |\mathcal{P}|\}, b \in \{1, \cdots, |\mathcal{S}|\}. \quad (4)$$

One could generally repeat this process a few rounds to iteratively refine the final instructions. That is, utilizing the automated-instructions generated at step τ denoted by $i'^{(\tau)}_{p_1 s_b} \in \mathcal{I}_{A^{(\tau)}}$, the iteration $(\tau + 1)$ generations can be formalized as

$$h_{\tau+1}(d(p_a), d(s_b), i'^{(\tau)}_{p_1 s_b}, \cdots, i'^{(\tau)}_{p_M s_b}) = i'^{(\tau+1)}_{p_a s_b}$$

Final iteration $\tau = T$ then yields the automated instructions $i'^{(T)}_{p_1 s_b} \in \mathcal{I}_{A^{(T)}}$ that will be utilized in the down-stream quality classification task.

This cascaded framework enables us to maintain instruction quality through hierarchical knowledge transfer and reduces manual effort from $O(K \times |\mathcal{S}|)$ to $O(M)$ by structured knowledge transfer between iterations. The pseudocode is provided in Alg. 1.

Model	Method	Precision	Recall	F1 score	Effort
Mixtral 8x7B	Baseline	50.08%	38.45%	43.50%	0
	CoT prompting	57.67%	44.98%	50.54%	10
	Human engineered	68.57%	51.24%	58.65%	308
	Auto prompt cascade	**70.13%**	**52.80%**	**60.24%**	**3**
Mixtral 8x22B	Baseline	60.27%	42.61%	49.92%	0
	CoT prompting	69.92%	61.32%	65.34%	10
	Human engineered	73.05%	66.01%	68.35%	308
	Auto prompt cascade	**81.46%**	**69.46%**	**74.98%**	**3**
DeepSeek R1 Distill Qwen 32B	Baseline	51.25%	75.33%	61.00%	0
	CoT prompting	57.50%	85.71%	68.83%	10
	Human engineered	66.10%	86.87%	75.09%	308
	Auto prompt cascade	**66.30%**	**88.21%**	**75.70%**	**3**
Claude 3.5 Sonnet	Baseline	66.00%	68.80%	67.37%	0
	CoT prompting	83.95%	74.64%	79.02%	10
	Human engineered	86.96%	80.34%	83.52%	308
	Auto prompt cascade	**92.04%**	**90.23%**	**91.13%**	**3**

Table 1: Performance comparison of different prompt generation methods and models for the correctness task on english. The full table is given in Table 5.

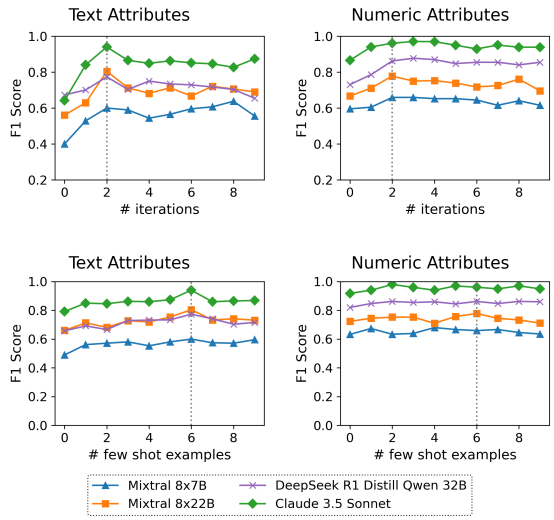

Figure 3: F1 score comparison on the incorrect class of three LLMs on the correctness task using Mixtral 8x7B, Mixtral 8x22B, DeepSeek R1 Distill Qwen 32B, and Claude 3.5 Sonnet. The top and bottom rows show F1 scores across the number of iterations and few shot examples respectively. Iteration 0 denotes performance of the CoT prompting. The dotted line represents best iteration across at least three of the four LLMs.

Language	Model	Prompting technique	Incorrectness		
			Precision	Recall	F1 score
Spanish	Mixtral 8x7B	Baseline	34.69%	8.95%	14.23%
		CoT prompting	56.92%	18.37%	27.78%
		Auto prompt cascade	**67.31%**	**19.27%**	**29.96%**
	Mixtral 8x22B	Baseline	41.18%	10.94%	17.29%
		CoT prompting	**62.75%**	16.67%	26.34%
		Auto prompt cascade	60.87%	**21.88%**	**32.19%**
German	Mixtral 8x7B	Baseline	39.06%	12.76%	19.24%
		CoT prompting	55.00%	22.22%	31.65%
		Auto prompt cascade	**61.97%**	**22.22%**	**32.71%**
	Mixtral 8x22B	Baseline	50.98%	13.07%	20.81%
		CoT prompting	**66.20%**	23.62%	34.82%
		Auto prompt cascade	60.64%	**28.64%**	**38.91%**
Italian	Mixtral 8x7B	Baseline	20.31%	7.93%	11.41%
		CoT prompting	62.20%	26.95%	37.61%
		Auto prompt cascade	**69.23%**	**30.36%**	**42.21%**
	Mixtral 8x22B	Baseline	44.62%	17.26%	24.89%
		CoT prompting	68.92%	30.36%	42.15%
		Auto prompt cascade	**78.46%**	**30.36%**	**43.78%**
French	Mixtral 8x7B	Baseline	28.00%	7.53%	11.87%
		CoT prompting	62.92%	**29.68%**	**40.33%**
		Auto prompt cascade	**78.12%**	26.32%	39.37%
	Mixtral 8x22B	Baseline	43.86%	13.16%	20.25%
		CoT prompting	**75.00%**	25.13%	37.65%
		Auto prompt cascade	68.97%	**31.41%**	**43.16%**

Table 2: Performance comparison of incorrectness metrics for different prompt generation methods on multilingual datasets. The full table is given in Table 6.

4 Experimental setup

4.1 Tasks and datasets

We evaluate our method on two product quality assessment tasks, correctness and applicability, using proprietary datasets sampled from an e-commerce catalog, which we describe below. The datasets were labeled by subject matter experts and they did a 10% random blind auditto ensure an accuracy of at least 98% of the evaluation set. [1]

Correctness: In this task, we are interested in identifying inconsistencies between UAs and SAs of a product. If an SA value is inconsistent with UAs, we call it *incorrect*, otherwise it is *correct*. For example, if a product description states that a jacket "keeps you dry in heavy downpours and harsh weather", but the SA "water resistance level" lists it as "water-resistant", the output would be *incorrect* since water-resistant materials only protect against light rain while waterproof materials fully block moisture.

We create a comprehensive human labeled English language dataset, covering 12,046 unique PC-SA combinations across 1,322 PCs and 1,410 SAs with both correct and incorrect labels. The dataset consists of 13,725 labels (12,148 correct, 1,577 incorrect). This is the primary task we are interested in, but we also perform experiments on other tasks that we describe below.

Multilingual correctness: To evaluate our framework's multilingual capabilities, we curate evaluation datasets across four major European languages. These datasets contain 1,014 labeled examples for Spanish (822 correct/192 incorrect), 1,102 for German (903 correct/199 incorrect), 1,011 for Italian (843 correct/168 incorrect), and 1,014 for French (823 correct/191 incorrect).

Applicability: In this task, we are interested in identifying the SAs that are not relevant (inapplicable) for a given product. For instance, in shoes PC, the SA "heel length" is inapplicable to a pair of sneakers while it is applicable to stilettos. Similar to the correctness-task, this is framed as a binary (Applicable/Inapplicable) classification task. Our English language dataset is comprised of 1,581 SAs where 70% of the instances belong to the Applicable class. We again utilize our auto prompt cascade's instructions on correctness task for applicability to demonstrate generalization across tasks.

4.2 Models and hyperparameters

We employ Claude 3.5 Sonnet (ANTHROPIC, 2024) to generate instructions for our auto prompt cascade and evaluate their effectiveness on four foundational LLMs: Mixtral 8x7B (Jiang et al., 2024a), Mixtral 8x22B (AI, 2024), DeepSeek R1

[1] To the best of our knowledge, there are no publicly available datasets for these tasks.

Model	Method	Precision	Recall	F1 score	Effort
Mixtral 8x7B	Baseline	52%	50.93%	51.45%	0
	CoT prompting	49.27%	55.46%	52.16%	10
	Auto prompt cascade	**54.83%**	**58.56%**	**56.63%**	**3**
Mixtral 8x22B	Baseline	62.25%	44.54%	51.92%	0
	CoT prompting	66.98%	43.92%	53.12%	10
	Auto prompt cascade	**69.30%**	**45.15%**	**54.62%**	**3**

Table 3: Performance comparison of different prompt generation methods/models for the applicability task. The full table is given in Table 7.

Distill Qwen 32B (Guo et al., 2025), and Claude 3.5 Sonnet.

To tune the hyperparameters - *iterations* T and *few-shot examples* M - for the auto prompt cascade, we construct two balanced datasets comprising text and numeric SAs for the English correctness task, each with 250 correct and 50 incorrect samples. We first determine the optimal value of T by conducting experiments for $T = 0 \ldots 6$, where $T = 0$ is a baseline without auto prompt cascade instructions, i.e. the CoT prompt. Using this optimal T, we then tune for M. To ensure robustness and generalization, we select the final hyperparameter values for T and M that yield optimal performance across a majority of tested LLMs.

4.3 Metrics and evaluation

For our binary classification tasks (Correct/Incorrect or Applicable/Inapplicable), we report precision, recall, and F1 score. We prioritize metrics on the negative class for hyperparameter tuning, reflecting our task's focus. To assess scalability, we measure *Effort*, the time in minutes required for manual prompt engineering per PC-SA.

We evaluate four methods: (1) a baseline using vanilla prompting, (2) CoT prompting (§3.2), (3) human-engineered instructions (§3.3), and (4) our auto-prompt cascade (§3.4). Due to the time-intensive nature of method (3), we apply it only to our primary task of English correctness. Similarly, to manage computational cost for the multilingual and applicability tasks, we report results on two LLMs: Mixtral 8x7B and Mixtral 8x22B. We show the prompt templates used in §A.

5 Results

Determining hyperparamaters: We determine the optimal number of iterations, T, using our hyperparameter tuning datasets (§4.2) for the correctness task. As plotted in Fig 3, F1 scores peak at T=2. The performance gain from T=1 to T=2 shows the model's ability to capture nuances through iterative refinement. For instance, the instruction for a walking stick's base material evolves from a generic definition at T=1: (*A base material for a walking stick refers to the primary material used to construct the base of the walking stick, which is typically wood, aluminum, carbon fiber, or other sturdy materials*) to a more contextualized instruction at T=2: (*Base material refers to the material that makes up the bottom part of a walking stick, which comes into contact with the ground and provides stability and traction when using the stick*). This improvement stems from the model first integrating domain-specific metadata at T=1, then performing SA-level refinement at T=2 for finer distinctions.

Our method is also robust to the number of few-shot examples, M. Fig 3 shows that with the optimal T=2, performance is steady across various values of M, with most models showing peak F1 score at M=6. Given these observations, we select T=2 and M=6 for our auto-prompt cascade in all subsequent experiments.

Correctness: As shown in Table 1, the auto prompt cascade consistently improves F1 scores for incorrect SA detection across all models. With Claude 3.5 Sonnet, it achieves a 91.13% F1 score, a 23.76% improvement over the baseline. We observe similar gains with DeepSeek R1 Distill Qwen 32B (61.00% to 75.70%), Mixtral 8x22B (from 49.92% to 74.98%), and Mixtral 8x7B (from 43.50% to 60.24%).

Our approach also significantly reduces human effort. The human-engineered method requires 308 minutes per PC-SA to cover only 2,388 PC-SAs, whereas our auto prompt cascade handles all 12,046 PC-SAs in just 3 minutes per PC-SA. This represents a 99% reduction in human time for superior performance. In comparison, the CoT approach requires 10 minutes per PC-SA but yields consistently lower performance with no PC-SA specific instructions.

Multilingual correctness: We use Mixtral 8x7B and Mixtral 8x22B to compare our auto prompt cascade with CoT and vanilla prompting approaches. Table 2 suggests that our approach improves the F1 score across all four languages - Spanish, German, Italian, and French by 10-30% over the baseline, demonstrating the cross-lingual generalization capability of our approach.

Applicability: Table 3 shows consistent improvements across F1 score for the Inapplicable

PC	SA (value)	CoT (abridged reasoning)	APC (abridged reasoning)
Walking stick	base material (rubber)	Incorrect: The product data mentions that the walking stick is made of 'Huangtang wood' and has a 'steel spike' and 'metal-reinforced rubber tip cover'. There is no mention of the walking stick being made of rubber material.	Correct: The product data mentions a 'metal-reinforced removable rubber tip cover' for the walking stick, which implies that the base material that comes into contact with the ground is rubber.

Table 4: Qualitative example contrasting CoT and auto-prompt cascade predictions. The table presents abridged reasoning segments that directly contributed to the correct decision, with complete rationales and further examples provided in §C.4.

class, from 51.45% to 56.63% for Mixtral 8x7B, and from 51.92% to 54.62% for Mixtral 8x22B, demonstrating the generalization capability of the automatically-generated instructions of the auto prompt cascade approach across correctness and applicability tasks.

Qualitative examples: To illustrate the effectiveness of our approach, we present a qualitative example where auto prompt cascade (APC) instructions helped the model correct its initial assessment error for the correctness task in Table 4. The table provides an abridged version of the reasoning that led to this correction, while full reasoning details and additional examples are discussed in §C.4. The PC is a walking stick and the value of the base material is rubber, which is correct in ground truth. The CoT prompt implicitly interprets base material as the material of the entire stick. Seeing "Huangtang wood" and a "steel spike," it concludes the test value "rubber" contradicts the product data and predicts Incorrect.

The APC instruction generated is *Base material refers to the material that makes up the bottom part of a walking stick, which comes into contact with the ground and provides stability and traction when using the stick.* With this disambiguation, the model's evidence retrieval shifts to the phrase "metal-reinforced removable rubber tip cover," and the rationale updates accordingly, yielding Correct. This example illustrates how the learned instruction resolves an attribute ambiguity at a PC level.

Cost: Our method generates instructions for each PC-SA combination offline and the generated instruction for the PC-SA is then used for all the items in the PC. Hence, the number of LLM inferences required for our method is linearly proportional to the number of PC-SAs (tens of thousands) and not the number of items in the catalog (hundreds of millions).

The final results in the paper needed only 20,506 instruction generation LLM calls. We generated the instructions using Claude 3.5 Sonnet, with an average of 2717 input tokens and 113 output tokens per LLM call. In comparison to CoT, our approach consumes the extra computational cost of $ 0.0097/PC-SA for creating the instructions, which is negligible. The inference costs on the downstream quality classification tasks for CoT and our approach are comparable as both use the same underlying LLM.

Statistical signifcance: We performed a statistical evaluation of the results, showing that the performance gains are statistically significant in the vast majority of cases. We assessed the significance using a paired subsampled bootstrap test with 5,000 class-balanced 80% draws without replacement, applying a plus-one correction. The detailed Δ F1 scores and the p values are shown in §C.5.

6 Conclusion and future work

We propose an auto-prompt cascade for assessing product quality with LLMs - the first scalable approach for multilingual catalog quality assessment. It expands a small set of human-crafted instructions (6) to over 12,000 PC-SA-specific prompts, reducing manual effort by 99%. Our cascade demonstrates improvement of classification performance across two tasks and generalizes to five languages and multiple LLMs. While applied to product quality, it can be extended to product categorization and attribute generation. This work also underscores the need for quantitative metrics to evaluate instruction quality directly, measuring clarity, consistency, and handling of edge cases.

7 Limitations

While our auto-prompt cascade demonstrates significant improvements, we acknowledge some limitations that offer avenues for future research.

Firstly, the evaluation of the generated instructions is currently extrinsic, measured only by their impact on downstream quality tasks. As noted in our conclusion, there could be a need for intrinsic, quantitative metrics to directly assess the quality of the instructions themselves - for instance, by measuring their clarity, consistency, and ability to

handle edge cases without relying on models run on downstream tasks.

Secondly, while we demonstrate strong performance on e-commerce quality tasks like correctness and applicability, the framework's effectiveness on other e-commerce tasks, such as attribute value generation or product categorization, remains an open question. Extending and validating the auto prompt cascade for these different tasks is a key direction for future work.

Finally, an additional limitation pertains to the proprietary nature of our dataset. To alleviate this, we provide a thorough description of the quality assessment tasks and a detailed statistical breakdown of the composition of the datasets.

References

Mistral AI. 2024. https://mistral.ai/news/mixtral-8x22b/.

Sarah Amsl, Iain Watson, Christoph Teller, and Steve Wood. 2023. Presenting products on websites–the importance of information quality criteria for online shoppers. *International Journal of Retail & Distribution Management*, 51(9/10):1213–1238.

ANTHROPIC. 2024. https://www.anthropic.com/news/introducing-claude.

Nick Baumann, Alexander Brinkmann, and Christian Bizer. 2024. Using llms for the extraction and normalization of product attribute values. In *Proceedings of the 28th European Conference on Advances in Databases and Information Systems (ADBIS 2024)*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ju-Seung Byun, Jiyun Chun, Jihyung Kil, and Andrew Perrault. 2024. Ares: Alternating reinforcement learning and supervised fine-tuning for enhanced multi-modal chain-of-thought reasoning through diverse ai feedback. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4410–4430.

Jiao Chen, Luyi Ma, Xiaohan Li, Nikhil Thakurdesai, Jianpeng Xu, Jason HD Cho, Kaushiki Nag, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2023. Knowledge graph completion models are few-shot learners: An empirical study of relation labeling in e-commerce with llms. *arXiv preprint arXiv:2305.09858*.

Zhu Cheng, Wen Zhang, Chih-Chi Chou, You-Yi Jau, Archita Pathak, Peng Gao, and Umit Batur. 2024. E-commerce product categorization with llm-based dual-expert classification paradigm. In *Proceedings of the 1st Workshop on Customizable NLP: Progress and Challenges in Customizing NLP for a Domain, Application, Group, or Individual (CustomNLP4U)*, pages 294–304.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint arXiv:2205.12548*.

Viet-Tung Do, Van-Khanh Hoang, Duy-Hung Nguyen, Shahab Sabahi, Jeff Yang, Hajime Hotta, Minh-Tien Nguyen, and Hung Le. 2024. Automatic prompt selection for large language models. *arXiv preprint arXiv:2404.02717*.

Shuzheng Gao, Chaozheng Wang, Cuiyun Gao, Xiaoqian Jiao, Chun Yong Chong, Shan Gao, and Michael Lyu. 2025. The prompt alchemist: Automated llm-tailored prompt optimization for test case generation. *arXiv preprint arXiv:2501.01329*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2023. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *arXiv preprint arXiv:2309.08532*.

Muhammad Usman Hadi, Rizwan Qureshi, Abbas Shah, Muhammad Irfan, Anas Zafar, Muhammad Bilal Shaikh, Naveed Akhtar, Jia Wu, Seyedali Mirjalili, and 1 others. 2023. Large language models: a comprehensive survey of its applications, challenges, limitations, and future prospects. *Authorea Preprints*.

Yutong He, Alexander Robey, Naoki Murata, Yiding Jiang, Joshua Williams, George J Pappas, Hamed Hassani, Yuki Mitsufuji, Ruslan Salakhutdinov, and J Zico Kolter. 2024. Automated black-box prompt engineering for personalized text-to-image generation. *arXiv preprint arXiv:2403.19103*, 2(5).

Pedro Herrero-Vidal, You-Lin Chen, Cris Liu, Prithviraj Sen, and Lichao Wang. 2024. Learning variant product relationship and variation attributes from e-commerce website structures. *arXiv preprint arXiv:2410.02779*.

Yining Huang, Keke Tang, and Meilian Chen. 2024. Leveraging large language models for enhanced nlp task performance through knowledge distillation and optimized training strategies. *arXiv preprint arXiv:2402.09282*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024a. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Ling Jiang, Keer Jiang, Xiaoyu Chu, Saaransh Gulati, and Pulkit Garg. 2024b. Hallucination detection in llm-enriched product listings. In *Proceedings of the Seventh Workshop on e-Commerce and NLP@ LREC-COLING 2024*, pages 29–39.

Satish Kathiriya, Mahidhar Mullapudi, and Rajath Karangara. 2023. Optimizing ecommerce listing: Llm based description and keyword generation from multimodal data. *Int. J. Sci. Res.(IJSR)*, 12:2123–2130.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Weize Kong, Spurthi Amba Hombaiah, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024. Prewrite: Prompt rewriting with reinforcement learning. *arXiv preprint arXiv:2401.08189*.

Lei Li, Yongfeng Zhang, Dugang Liu, and Li Chen. 2023. Large language models for generative recommendation: A survey and visionary discussions. *arXiv preprint arXiv:2309.01157*.

Jun Lv and Xuan Liu. 2022. The impact of information overload of e-commerce platform on consumer return intention: Considering the moderating role of perceived environmental effectiveness. *International Journal of Environmental Research and Public Health*, 19(13):8060.

Xilai Ma, Jing Li, and Min Zhang. 2023. Chain of thought with explicit evidence reasoning for few-shot relation extraction. *arXiv preprint arXiv:2311.05922*.

Reza Yousefi Maragheh, Lalitesh Morishetti, Ramin Giahi, Kaushiki Nag, Jianpeng Xu, Jason Cho, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2023. Llm-based aspect augmentations for recommendation systems.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40.

Athanasios N Nikolakopoulos, Swati Kaul, Siva Karthik Gade, Bella Dubrov, Umit Batur, and Suleiman Ali Khan. 2023. Sage: Structured attribute value generation for billion-scale product catalogs. *arXiv preprint arXiv:2309.05920*.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. 2024. Optimizing instructions and demonstrations for multi-stage language model programs. *arXiv preprint arXiv:2406.11695*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Md Omar Faruk Rokon, Andrei Simion, Weizhi Du, Musen Wen, Hong Yao, and Kuang-chih Lee. 2024. Enhancement of e-commerce sponsored search relevancy with llm.

Mauricio Sadinle, Karim Bouyarmane, Grant Galloway, Shioulin Sam, Changhe Yuan, and Ismail Tutar. 2022. Semi-automated estimation of weighted rates for e-commerce catalog quality monitoring.

Oliver Schmidts, Bodo Kraft, Marvin Winkens, and Albert Zündorf. 2020. Catalog integration of low-quality product data by attribute label ranking. In *DATA*, pages 90–101.

Dilara Soylu, Christopher Potts, and Omar Khattab. 2024. Fine-tuning and prompt optimization: Two great steps that work better together. *arXiv preprint arXiv:2407.10930*.

Gaike Wang, Xin Ni, Qi Shen, and Mingxuan Yang. 2024. Leveraging large language models for context-aware product discovery in e-commerce search systems. *Journal of Knowledge Learning and Science Technology ISSN: 2959-6386 (online)*, 3(4):300–312.

Shuyang Wang, Somayeh Moazeni, and Diego Klabjan. 2025. A sequential optimal learning approach to automated prompt engineering in large language models. *arXiv preprint arXiv:2501.03508*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Yaqing Wang, Yifan Ethan Xu, Xian Li, Xin Luna Dong, and Jing Gao. 2020. Automatic validation of textual attribute values in e-commerce catalog by learning with limited labeled data. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2533–2541.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yue Zhang, Hongliang Fei, Dingcheng Li, and Ping Li. 2022a. Promptgen: Automatically generate prompts using generative models. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 30–37.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022b. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

Arkaitz Zubiaga. 2024. Natural language processing in the era of large language models.

A Prompt templates

The following subsections contain the prompt templates that we use for our experiments.

A.1 Baseline prompt template

We use the prompt template shown in Figure 4 for the baseline experiment as detailed in §4.3. We use the same baseline prompt for all PC-SA pairs. Given below is an example of the template for the SA - base material. This template instructs the model to adopt an auditor persona and perform a binary classification (*Correct* or *Incorrect*). Note that it defines *Correct* as the absence of a contradiction, rather than requiring explicit confirmation, which is inline with our correctness problem defined in §4.1.

A.2 CoT prompt template

For the CoT experiment as detailed in §4.3, we use different prompts for different groups of SAs. The prompt template shown in Figure 5 is an example used for the attribute *base material* in the *walking stick* product category.

Unlike the baseline, this CoT prompt is highly structured with explicit sections for introduction, data, rules, test value, and output format. The key component is the *Rules* section, which contains a placeholder for a sequence of reasoning steps that guide the LLM make accurate classifications. This design encourages the model to "think systematically" by following a prescribed logical flow before making its final assessment, aiming to improve accuracy on nuanced and complex cases.

A.3 Auto prompt cascade prompt template

Finally, for our proposed auto-prompt cascade method, we augment the CoT prompt with an additional *Instruction* field. This section contains the nuanced, domain-specific definition that is automatically generated by our cascade. The prompt template shown in Figure 6 illustrates this for the base material attribute of a walking stick. Notice it is identical to the CoT template in §A.2, with the exception of the crucial auto-generated instruction.

This final template directly addresses the core challenge of domain-specific knowledge gaps. By inserting the auto-generated instruction, we provide the LLM with a precise, contextual definition for the PC-SA pair. Furthermore, the instruction forces the model to ground its reasoning in this new information, making its decision making process more transparent and aligned with the specific requirements of the e-commerce catalog.

B Hyperparameter Analysis

This section presents a detailed analysis of the hyperparameter tuning experiments for our auto-prompt cascade method. As introduced in §4.2, Figure 7 illustrates the precision, recall, and F1 score of the incorrect class across multiple iterations, T, of the auto-prompt cascade. Iteration 0 represents the baseline performance obtained with standard CoT prompting, prior to the application of the auto-prompt cascade. The dotted line in each subplot denotes the empirically determined optimal iteration, defined as the earliest iteration where at least three of the four evaluated LLMs achieve their peak performance for the corresponding metric. Notably, our method consistently demonstrates optimal performance around the second iteration, indicating rapid convergence and highlighting the efficiency of the auto-prompt cascade in improving model performance.

After determining the optimal number of iterations, $T = 2$, we proceed with hyperparameter tuning on the same datasets to ascertain the optimal number of few-shot examples, M. Figure 8 illustrates the impact of varying M on the precision, recall, and F1 score for the incorrect class. Iteration 0 in this context signifies instruction generation based solely on the PC and SA definitions, without any few-shot examples. The dotted line in each subplot marks the optimal selection of few-shot examples, corresponding to the earliest point where the majority of the evaluated LLMs exhibit peak performance. Observing the results presented in Figure 7, it is evident that the majority of the subplots (5 out of 6) demonstrate optimal performance for the LLMs when $M = 6$ few-shot examples are utilized. This consistent trend across diverse metrics and models strongly indicates that $M = 6$ represents the most effective configuration for instruction generation within our auto-prompt cascade method. Consequently, we adopted this value for M in our experiments involving the auto-prompt cascade.

C Full results

In this section, we present the full results of the following tables - Table 1, Table 2, and Table 3. In these tables, we show the precision/recall/F1 score metrics on the positive class as well ("cor-

```
You are an auditor for an e-commerce store. You are given a product and its
data below. You will also be given a test value for 'base material'.

<Product data goes here.>

The test value for 'base material' is 'rubber'. Based on the given
product data, you have to say if the test value is 'Correct' or 'Incorrect'. If
the product data does not contradict the given value, your prediction should
be 'Correct'.

Output the results in the following output format.
<Output format>
```

Figure 4: Baseline prompt template for the attribute *base material*.

```
### Introduction:
You are an auditor for an e-commerce store. You are given a product and its
data below. You will also be given a test value for 'base material'.
Please classify the value as 'Correct' or 'Incorrect' based on the rules given
below.

### Product data:
Given below is the product data.
<Product data goes here.>

### Rules:
To ensure accurate predictions, adhere to the following rules in sequence and
think systematically before responding:
<CoT rules go here.>

### Test value:
Now verify the test value of the attribute 'base material': 'rubber'.

### Output format:
Output the results in the following output format.
<Output format>
```

Figure 5: CoT prompt template for the product category *walking stick* and attribute *base material*.

rect" for the correctness task and "applicable" for the applicability task). We also provide the numbers for accuracy on the respective datasets and the numbers of unique PC-SA combinations which had instructions generated for a particular experiment.

C.1 English correctness results

As detailed in section 4.1, correctness in english is our main dataset with 13,725 labels (12,148 correct, 1,577 incorrect) across 12,046 unique PC-SA combinations. Table 5 is the full version of Table 1. It presents the performance comparison of four LLMs, Mixtral 8x7B, Mixtral 8x22B, DeepSeek R1 Distill Qwen 32B, and Claude 3.5 Sonnet, across four prompting methods: baseline, Chain-of-Thought (CoT) prompting, human-engineered prompts, and our proposed auto-prompt cascade. Across all models and metrics, the auto-prompt cascade consistently outperforms other methods. For instance, on Mixtral 8x22B, it achieves the high-

```
### Introduction:
You are an auditor for an e-commerce store. You are given a product and its
data below. You will also be given a test value for 'base material'.
Go through the instruction to understand what 'base material' means in context
of this product. Please classify the value as 'Correct' or 'Incorrect' based
on the rules given below.

### Product data:
Given below is the product data.
<Product data goes here.>

### Rules:
To ensure accurate predictions, adhere to the following rules in sequence and
think systematically before responding:
<CoT rules go here.>

### Test value:
Now verify the test value of the attribute 'base material': 'rubber'.

### Instruction:
Here is some additional information about 'base material' to help you make
highly accurate classifications.
In your reasoning, explain how you applied this information to reach your
conclusion.

Base material refers to the material that makes up the bottom part of a
walking stick, which comes into contact with the ground and provides stability
and traction when using the stick.

### Test value:
Now verify the test value of the attribute 'base material': 'rubber'.

### Output format:
Output the results in the following output format.
<Output format>
```

Figure 6: Auto prompt cascade prompt template for the product category *walking stick* and attribute *base material*. We add the auto generated instruction to the walking stick-base material PC-SA pair to the CoT prompt template.

est incorrectness F1 score of 74.98% and correctness F1 score of 97.02%, with an overall accuracy of 94.37%, surpassing both CoT prompting and human-engineered prompts. Notably, the cascade scales to 12,046 unique PC-SA combinations with just 3 minutes of human effort per unique PC-SA combination, compared to 5.1 hours (308 minutes) for human-engineered prompts. This trend holds consistently across other models, with Claude 3.5 Sonnet reaching a correctness F1 of 98.86% and an accuracy of 97.62%.

When comparing model performance under the same prompting method, larger models consistently outperform smaller ones. For instance, under CoT prompting, Mixtral 8x22B outperforms Mixtral 8x7B on correctness F1 score (95.81% vs. 94.33%), incorrectness F1 score (65.34% vs 50.54%), and accuracy (92.21% vs. 89.50%). Likewise, human-engineered prompts show progressively better results on larger models, with Claude 3.5 Sonnet achieving the highest correctness F1 of 97.95% under this setting. Under the auto-prompt cas-

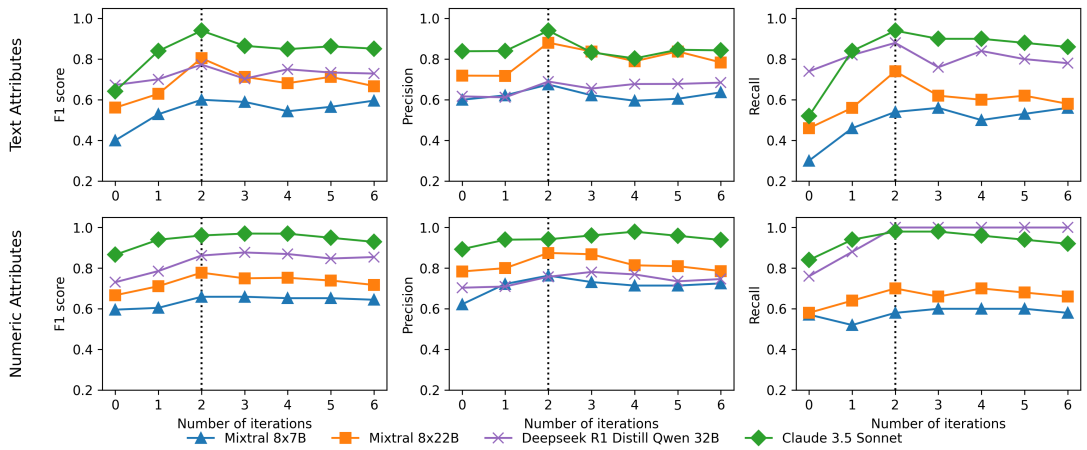

Figure 7: Performance comparison on the incorrect class of four LLMs across the number of iterations used for instruction generation on the correctness task using Mixtral 8x7B, Mixtral 8x22B, DeepSeek R1 Distill Qwen 32B, and Claude 3.5 Sonnet. The top and bottom rows show performance on text and numeric attributes respectively. Iteration 0 denotes performance before the auto-prompt cascade. The dotted line represents best iteration across majority of the LLMs.

cade, Claude 3.5 Sonnet consistently leads with the best overall performance (correctness F1 98.86%, accuracy 97.62%), followed by Mixtral 8x22B and DeepSeek R1 Distill Qwen 32B. Interestingly, while DeepSeek R1 Distill Qwen 32B lags behind on baseline and CoT prompting, its performance under the auto-prompt cascade improves substantially, achieving a correctness F1 of 96.24% and accuracy of 93.17%, indicating that prompt quality has an outsized impact on lower-capacity models. These results demonstrate that the auto-prompt cascade consistently boosts performance across model sizes and architectures, narrowing the gap between smaller and larger models while dramatically reducing manual effort.

C.2 Multilingual correctness results

We mention in §4.1 that we also test our auto prompt cascade on four european language datasets - Spanish, German, Italian, and French. These datasets contain 752, 799, 734, and 793 PC-SA pairs respectively. Table 6, the full version of Table 2, reports the precision, recall, F1 score, and accuracy for both correctness and incorrectness classifications using Mixtral 8x7B and Mixtral 8x22B across baseline, CoT prompting, and auto-prompt cascade methods. In all cases, the auto-prompt cascade outperforms both baseline and CoT prompting on the key metrics of Incorrectness F1 score, Correctness F1 score, and Accuracy. For example, in Spanish, Mixtral 8x7B's Incorrectness F1 score improves from 14.23% (baseline) and 27.78% (CoT) to 29.96% with the auto prompt cascade, while Correctness F1 rises from 88.37% to 90.20%, and accuracy increases from 79.52% to 81.37%.

When comparing models within each prompting technique, Mixtral 8x22B consistently outperforms Mixtral 8x7B on Incorrectness F1, Correctness F1, and Accuracy for both baseline and CoT prompting across all languages. For instance, under CoT prompting in Italian, Mixtral 8x22B achieves an Incorrectness F1 score of 42.15%, Correctness F1 score of 92.13%, and accuracy of 83.18%, compared to 37.61%, 91.44%, and 82.32% for 8x7B, respectively. However, the auto-prompt cascade substantially narrows this performance gap. In French, Mixtral 8x7B with the cascade achieves an Incorrectness F1 of 39.37%, Correctness F1 score of 91.23%, and accuracy of 81.59%, approaching the CoT-prompted 8x22B values of 43.16%, 90.97%, and 81.36%. In several cases, the cascade-augmented 8x7B even outperforms CoT-prompted 8x22B, such as in Italian where it reaches 42.21% Incorrectness F1 score, 92.20% Correctness F1 score, and 82.90% accuracy. These results reaffirm the cascade's strong generalizability and its ability to consistently improve both correctness and error detection across models and languages while reducing manual effort.

C.3 English applicability results

We also show the full results table of the applicability task (introduced in §4.1). Table 7 is an expanded version of Table 3, reporting metrics on our

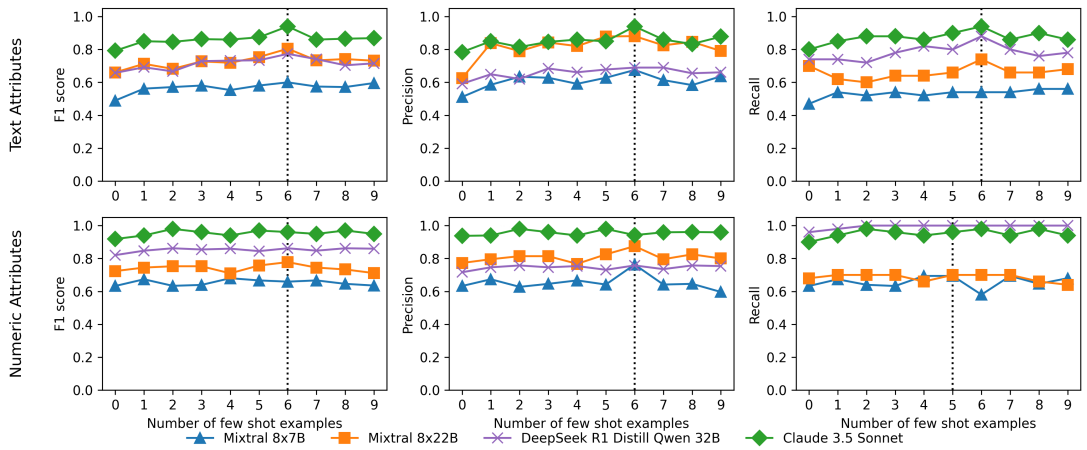

Figure 8: Performance comparison on the incorrect class across the number of few shot examples used for instruction generation on the correctness task, using Mixtral 8x7B, Mixtral 8x22B, DeepSeek R1 Distill Qwen 32B, and Claude 3.5 Sonnet. The top and bottom rows show performance on text and numeric attributes respectively. Iteration 0 signifies instruction generation using only the PC and SA definitions. The dotted line represents optimal selection of few-shot examples across majority of the LLMs.

Model	Method	Incorrectness			Correctness			Accuracy	Unique PC-SA	Effort
		Precision	Recall	F1 score	Precision	Recall	F1 score		Count	
Mixtral 8x7B	Baseline	50.08%	38.45%	43.50%	92.23%	95.01%	93.60%	88.18%	0	0
	CoT prompting	57.67%	44.98%	50.54%	93.01%	95.68%	94.33%	89.50%	0	10
	Human engineered	68.57%	51.24%	58.65%	93.89%	96.96%	95.40%	91.40%	2388	308
	Auto prompt cascade	**70.13%**	**52.80%**	**60.24%**	**94.05%**	**97.07%**	**95.54%**	**91.66%**	12046	**3**
Mixtral 8x22B	Baseline	60.27%	42.61%	49.92%	92.82%	96.35%	94.55%	89.87%	0	0
	CoT prompting	69.92%	61.32%	65.34%	95.06%	96.58%	95.81%	92.21%	0	10
	Human engineered	73.05%	66.01%	68.35%	95.64%	96.84%	96.24%	92.98%	2388	308
	Auto prompt cascade	**81.46%**	**69.46%**	**74.98%**	**96.11%**	**97.95%**	**97.02%**	**94.37%**	12046	**3**
DeepSeek R1 Distill Qwen 32B	Baseline	51.25%	75.33%	61.00%	96.59%	90.70%	93.55%	88.62%	0	0
	CoT prompting	57.50%	85.71%	68.83%	93.02%	91.78%	94.80%	90.77%	0	10
	Human engineered	66.10%	86.87%	75.09%	98.22%	94.21%	96.17%	93.07%	2388	308
	Auto prompt cascade	**66.30%**	**88.21%**	**75.70%**	**98.40%**	**94.17%**	**96.24%**	**93.17%**	12046	**3**
Claude 3.5 Sonnet	Baseline	66.00%	68.80%	67.37%	95.93%	95.40%	95.66%	92.03%	0	0
	CoT prompting	83.95%	74.64%	79.02%	96.75%	98.15%	97.44%	95.13%	0	10
	Human engineered	86.96%	80.34%	83.52%	97.47%	98.44%	97.95%	96.04%	2388	308
	Auto prompt cascade	**92.04%**	**90.23%**	**91.13%**	**98.74%**	**98.99%**	**98.86%**	**97.62%**	12046	**3**

Table 5: Performance comparison of different prompt generation methods and models for the correctness task on the english language.

English dataset spanning 1,310 unique PC-SA combinations. Across both Mixtral 8x7B and 8x22B, the auto-prompt cascade consistently outperforms baseline and CoT prompting on Inapplicable F1, Applicable F1, and Accuracy. For Mixtral 8x7B, the cascade improves Inapplicable F1 from 51.45% (baseline) and 52.16% (CoT) to 56.63%, Applicable F1 from 78.83% and 76.84% to 79.85%, and Accuracy from 70.52% to 72.49%. Similar trends hold for Mixtral 8x22B, where the cascade increases Inapplicable F1 from 51.92% (baseline) and 53.12% (CoT) to 54.62%, Applicable F1 from 82.83% and 83.91% to 84.63%, and Accuracy from 74.70% to 77.04%. As with other tasks, the cascade achieves these improvements while scaling to all 1,310 PC-SA combinations with just 3 minutes of human effort per combination, confirming its efficiency and generalizability.

C.4 Qualitative results

In Table 8, we present qualitative examples that demonstrate the impact of integrating PC-SA instructions within our auto prompt cascade framework for the correctness task in english. These cases illustrate how targeted, context-aware instructions can substantially improve model performance by guiding the model to attend to PC specific nuances that generic prompts may overlook. By comparing the model's initial CoT predictions with those informed by PC-SA instructions, we highlight the value of structured, domain-adapted guidance in reducing assessment errors and enhancing

Model	Prompting technique	Incorrectness			Correctness			Accuracy	Unique PC-SAs with inst.	Effort
		Precision	Recall	F1 score	Precision	Recall	F1 score			
Spanish										
Mixtral 8x7B	Baseline	34.69%	8.95%	14.23%	81.83%	96.05%	88.37%	79.52%	0	0
	CoT prompting	56.92%	18.37%	27.78%	83.51%	96.56%	89.56%	80.40%	0	10
	Auto prompt cascade	**67.31%**	**19.27%**	**29.96%**	**83.61%**	**97.91%**	**90.20%**	**81.37%**	752	**3**
Mixtral 8x22B	Baseline	41.18%	10.94%	17.29%	82.11%	96.32%	88.65%	80.04%	0	0
	CoT prompting	**62.75%**	16.67%	26.34%	83.28%	**97.67%**	89.90%	80.85%	0	10
	Auto prompt cascade	60.87%	**21.88%**	**32.19%**	**84.03%**	96.69%	**89.92%**	**81.05%**	752	**3**
German										
Mixtral 8x7B	Baseline	39.06%	12.76%	19.24%	81.55%	95.09%	87.80%	78.81%	0	0
	CoT prompting	55.00%	22.22%	31.65%	83.39%	95.55%	89.06%	79.05%	0	10
	Auto prompt cascade	**61.97%**	22.22%	**32.71%**	**83.49%**	**96.65%**	**89.59%**	**79.88%**	799	**3**
Mixtral 8x22B	Baseline	50.98%	13.07%	20.81%	81.96%	**96.92%**	88.81%	**80.40%**	0	0
	CoT prompting	**66.20%**	23.62%	34.82%	83.85%	97.05%	**89.97%**	80.53%	0	10
	Auto prompt cascade	60.64%	**28.64%**	**38.91%**	**84.53%**	95.45%	89.66%	80.24%	799	**3**
Italian										
Mixtral 8x7B	Baseline	20.31%	7.93%	11.41%	83.68%	93.82%	88.46%	79.58%	0	0
	CoT prompting	62.20%	26.95%	37.61%	87.04%	96.31%	91.44%	82.32%	0	10
	Auto prompt cascade	**69.23%**	**30.36%**	**42.21%**	**87.35%**	**97.62%**	**92.20%**	**82.90%**	734	**3**
Mixtral 8x22B	Baseline	44.62%	17.26%	24.89%	85.26%	95.71%	90.18%	82.64%	0	0
	CoT prompting	68.92%	30.36%	42.15%	87.51%	97.27%	92.13%	83.18%	0	10
	Auto prompt cascade	**78.46%**	30.36%	**43.78%**	**87.63%**	**98.34%**	**92.68%**	**84.08%**	734	**3**
French										
Mixtral 8x7B	Baseline	28.00%	7.53%	11.87%	81.72%	95.53%	88.09%	79.01%	0	0
	CoT prompting	62.92%	**29.68%**	**40.33%**	**85.54%**	95.98%	90.46%	80.48%	0	10
	Auto prompt cascade	**78.12%**	26.32%	39.37%	85.12%	**98.28%**	**91.23%**	**81.59%**	793	**3**
Mixtral 8x22B	Baseline	43.86%	13.16%	20.25%	82.74%	96.11%	88.93%	80.55%	0	0
	CoT prompting	**75.00%**	25.13%	37.65%	84.93%	**98.05%**	**91.02%**	81.24%	0	10
	Auto prompt cascade	68.97%	**31.41%**	**43.16%**	**85.87%**	96.72%	90.97%	**81.36%**	793	**3**

Table 6: Performance comparison of different prompt generation methods on multilingual correctness datasets.

Model	Method	Inapplicable			Applicable			Accuracy	Unique PC-SA Count	Effort
		Precision	Recall	F1 score	Precision	Recall	F1 score			
Mixtral 8x7B	Baseline	52%	50.93%	51.45%	78.48%	**79.20%**	78.83%	70.52%	0	0
	CoT prompting	49.27%	55.46%	52.16%	79.13%	74.73%	76.84%	68.82%	0	10
	Auto prompt cascade	**54.83%**	**58.56%**	**56.63%**	**81.09%**	78.65%	**79.85%**	**72.49%**	1310	**3**
Mixtral 8x22B	Baseline	62.25%	44.54%	51.92%	78.20%	88.05%	82.83%	74.70%	0	0
	CoT prompting	66.98%	43.92%	53.12%	78.46%	90.42%	83.91%	76.15%	0	10
	Auto prompt cascade	**69.30%**	**45.15%**	**54.62%**	**78.97%**	**91.15%**	**84.63%**	**77.04%**	1310	**3**

Table 7: Performance comparison of different prompt generation methods and models for the applicability task

interpretability in product quality tasks.

In the first example, for a walking stick's base material, the initial CoT reasoning fails to link a *metal-reinforced removable rubber tip cover* to the SA value *rubber*. Our auto generated instruction explicitly defines the base material as the component that contacts the ground and provides stability and traction. This PC specific clarification enables the model to correctly infer that the rubber tip, which interacts with the ground, justifies the SA value *rubber* despite the main body being *Huantang wood*. The corrected assessment shows how such context-sensitive instructions can resolve ambiguities arising from indirect evidence in product descriptions.

In the second example, for the *age range* SA of a pet food product, the initial model prediction was correct due to a lack of explicit age information in the product data. However, the PC-SA instruction emphasizes the need to map age range attributes to the intended life stage of the animal, as inferred from product descriptions. This instruction prompted the model to scrutinize the product title and bullet points, which referenced *puppy* and *rapidly growing puppies*, indicating the product targets the puppy growth stage rather than *young adult*. Consequently, the model revised its assessment, identifying the mismatch between the test value and the actual target demographic. This example underscores how PC-SA instructions can surface implicit cues in product data, enabling more accurate and contextually grounded SA validation.

Product category	Walking stick	Pet Food
Attribute	base material	age range
Attribute value	rubber	young adult
CoT pred.	Incorrect	Correct
CoT reasoning	The product data mentions that the walking stick is made of 'Huangtang wood' and has a 'steel spike' and 'metal-reinforced rubber tip cover'. There is no mention of the walking stick being made of rubber material. Therefore, the test value 'Rubber' for the attribute 'base material' contradicts the facts stated in the product data.	The product data does not contain any information about the intended age range for this dog food product. The title and description focus on nutritional details for growing puppies, but do not specify an age range. Without clear evidence to confirm or contradict the test value of 'Young Adult', the prediction must be 'Correct'.
Auto prompt cascade PC-SA instruction	Base material refers to the material that makes up the bottom part of a walking stick, which comes into contact with the ground and provides stability and traction when using the stick.	Age range indicates the intended or appropriate age group for which the pet food product is suitable. Pet food is designed to provide proper nutrition for animals at different life stages, and the age range helps ensure that the product meets the specific dietary needs of pets based on their age. It guides users to select the most appropriate age range for their pet, ensuring they receive the right nutritional balance for their growth and development stage.
Auto prompt cascade pred.	Correct	Incorrect
Auto prompt cascade reasoning	The product data mentions a 'metal-reinforced removable rubber tip cover' for the walking stick, which implies that the base material that comes into contact with the ground is rubber. Therefore, the test value 'Rubber' for 'base.material' is consistent with the information provided in the product data.	The product title 'Complete And Balanced Dry Dog Food For Growing Puppies 20Lb' and the bullet points mentioning 'puppy' and 'rapidly growing puppies' clearly indicate that this product is intended for puppies or young dogs in their growth stage. The test value 'Young Adult' contradicts this information as it refers to a different life stage beyond the puppy/growing phase. Therefore, the test value is incorrect for this specific product.

Table 8: Examples of improved correctness decisions with automated PC-SA specific instructions.

Model	APC vs method	Mean Δ F1	p-value
Mixtral 8×7B	Baseline	0.1695	<0.002
	CoT	0.0986	<0.002
	Human engineered	0.0173	0.0353
Mixtral 8×22B	Baseline	0.2503	<0.002
	CoT	0.0960	<0.002
	Human engineered	0.0556	<0.002
DeepSeek R1 Distill Qwen 32B	Baseline	0.1465	<0.002
	CoT	0.0688	<0.002
	Human engineered	0.0061	0.1615
Claude 3.5 Sonnet	Baseline	0.2377	<0.002
	CoT	0.1214	<0.002
	Human engineered	0.0762	<0.002

Table 9: Statistical evaluation of APC improvements over baseline, CoT, and human-engineered prompts.

C.5 Statistical significance

Table 9 reports the statistical significance of the improvements of auto prompt cascade (APC) over baseline, CoT, and human-engineered prompts across multiple models. We perform these tests on our primary task of english correctness. We measure the significance using a paired subsampled bootstrap test with 5,000 class-balanced 80% draws without replacement and a plus-one correction.

We compute the Δ F1 scores and the p-values between APC and each method across all the four models - Mixtral 8x7B, Mixtral 8x22B, DeepSeek R1 Distill Qwen 32B, and Claude 3.5 Sonnet. We observe consistent and significant gains in nearly all comparisons, with p-values below 0.002 in the majority of cases. These results confirm that the performance improvements are robust and not attributable to random variation.