Applied Computing and Intelligence

https://www.aimspress.com/journal/aci

Applied Computing and Intelligence
6(2): 173–185.
DOI: 10.3934/aci.2026010
Received: 01 June 2026
Revised: 19 August 2026
Accepted: 19 August 2026
Published: 28 August 2026

Research article

Using the consistency of LLM for better attribute value extraction

Eetu Kyyrö and Pasi Fränti*

School of Computing, University of Eastern Finland, Joensuu, Finland

* **Correspondence:** Email: franti@cs.uef.fi.

Academic Editor: Chic-Cheng Hung

Abstract: Product attributes can be extracted from free-form text content using a large language model (LLM) with high precision. We propose repeating the extraction several times and selecting the most frequent extraction. We measure consistency as the percentage of the most frequent extraction and accept only the most consistent extractions. This approach reduces error rate from 0.5% with the baseline LLM to 0.2%, and, more importantly, eliminates all hallucinations (specificity increases from 99.1% to 100%). The method could also be used for LLM-assisted extraction where a human verifies the results by focusing on the less consistent extractions, and those attributes that the LLM did not find values for.

Keywords: Large language models; LLM; attribute value extraction; product data management; GPT-4o

1. Introduction

Product data management (PDM) refers to processes for systematic maintenance of product information [1]. The information can be used in e-commerce, warehouse management, and marketing. The aim is to have accurate, up-to-date, and consistent information across various platforms and media.

The information includes not only the product name and text description, but also more detailed, precise technical information, such as color, size, or nutritional content, for a food product. For example, a customer is looking for new wheels matching his car and wants to see only the products of the correct size and type, e.g., 175/70 winter tires, see Figures 1 and 2.

The typical workflow is to extract information manually from a product description, whether in a PDF or a paper brochure. We need to locate the relevant attributes, extract their values, and input them into the product database. It is desired that the attributes and their values are stored

systematically for the given product category with high coverage. If a product misses essential value, it can be missed in search results, leading to reduced sales.

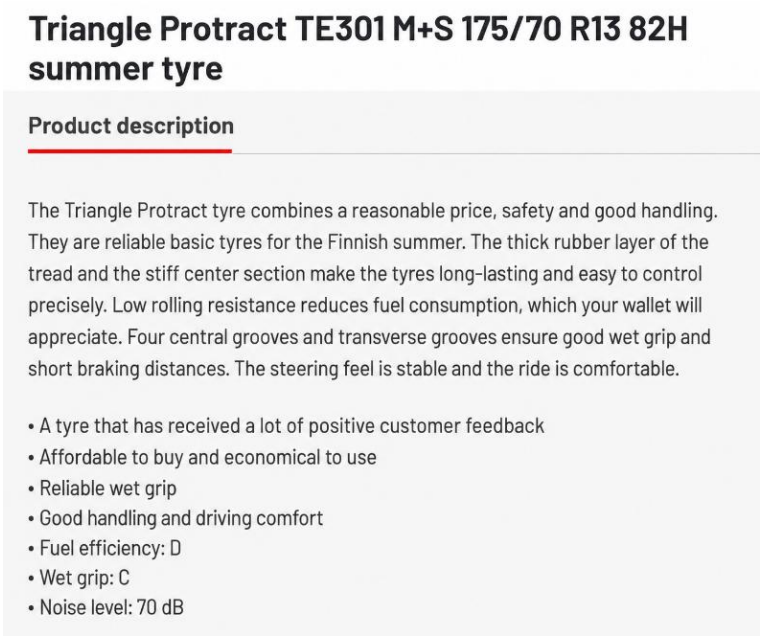


Figure 1. Example of a product name and description (car tire).

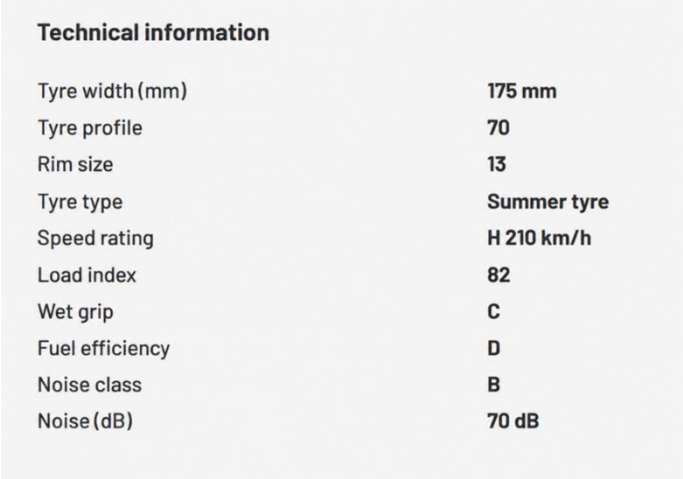


Figure 2. Example of structured product information (car tire) shown on the product web page.

Another important requirement is that the stored attribute values are correct. Wrong values can lead to product returns, causing financial losses but, more importantly, degrading brand value.

However, manual processing can be time-consuming, prone to human error, and poorly scalable for large product ranges. The amount of work can be significant, especially if a new attribute is added to an existing product category. For example, if we want to add the share of recycled materials in car tires, employees must review all products and record the correct information for each.

Automatic information extraction has been studied since the 1960s, with the first commercial applications appearing in the 1980s. In recent decades, machine learning-based methods have become more common, especially in recent years, with development focusing on the application of large language models (LLMs). They have been extensively studied for information extraction through

prompt engineering, example-based methods, chain-of-thought reasoning, and fine-tuning. However, we have not seen consistent use of repeated extractions.

LLMs are suitable, especially for summarization. They can generate responses that appear semantically coherent, even if they are not always factually correct. One key challenge is *hallucination*, where an LLM produces information that does not exist. This phenomenon highlights the need to develop methods to assess the reliability of information produced by LLMs.

In this paper, we use large language models for extracting product attributes from product names and descriptions. We compare the extracted attributes with existing, human-recorded attribute-value pairs in real-system data as a benchmark. We propose using the consistency of LLM output to improve accuracy. LLM generates multiple extractions for each product attribute, and we measure extraction consistency as the frequency with which the system provides the same attribute value. The hypothesis is that higher consistency leads to higher accuracy.

We aim to answer the following research questions:

- How accurate are the predictions?
- Does accuracy improve when the process is repeated?
- Can consistency be used to measure reliability?

Additional questions are:

- Is the method cost-efficient compared to manual work?
- What are the risks and limitations of the automated extraction?

The extraction is a two-objective optimization. We want high coverage by extracting as many attribute values as possible. We refer to this as high recall. However, the more we accept uncertain extraction results, the more we will have incorrect extraction results and even hallucinated attribute values. We refer to this as precision. Tightening the consistency threshold improved accuracy and reduced hallucination, but decreased coverage.

2. Product information extraction

On online shopping platforms, incorrect or missing product information can cause customer dissatisfaction, increased product returns, and damage a brand's reputation. According to [2], 54% of consumers have returned a product due to incorrect product information, 63% have canceled their purchase decision, and 63% have considered abandoning the brand due to a bad customer experience. In other words, product returns are not the only consequence of incorrect information; they can also affect the entire customer relationship and damage the brand image. According to [3], high-quality product information can significantly increase sales and conversions. This emphasizes the need for high-quality, consistent product data management.

Finding the products in an online shop requires searches based on attributes like correct size and type, such as 205/55 R16 winter tires. Product description may contain ambiguous and context-dependent information, like the word *black*, which may refer to the main color of the jacket or coat lining. Simple text searches may therefore yield inaccurate, misleading, or missing results.

2.1. Information extraction

Product information is often available in free-form text or included in the product name, such as *black winter jacket with hood*. Information in such a form cannot be used for exact queries, but it should be stored in a structured form, e.g., as attribute-value pairs such as *color-yellow*. The main challenge is to convert free-form text into attribute-value pairs.

In 2010s, classical rule-based methods were replaced by machine learning methods such as neural networks, sequence labeling, and bidirectional LSTM networks as they reduced the need for hand-crafted rules [4]. Transformer architecture using pre-trained language models was introduced in [5]. Particularly popular became BERT [6], which tailored the same language model to different NLP tasks, including RoBERTa [7], particularly for information extraction.

Variations in language make information extraction challenging. Synonyms, polysemes, and pronouns require understanding of the context. For example, "*A hooded winter coat made of black fabric*" contains the same information as a "*Black winter jacket with a hood*" but in a different form. This kind of variation is challenging not only to classical rule-based systems but also to machine learning-based methods.

2.2. Large language models (LLMs)

A large language model refers to a large neural network trained using a huge text corpus [8]. They are based on a transformer architecture [5] and revolutionized natural language processing by replacing recursive neural networks (RNNs) and convolutional neural networks with a self-attention mechanism. It also enables more powerful parallel processing and the handling of longer-term dependencies.

Recent examples of large generative models include GPT-4o and Meta Llama 2, but they evolve fast. Unlike traditional models, such as ELMo and BERT, the generative models can work with only a few examples (few-shot) or even without examples (zero-shot) [8]. With the help of easy-to-use interfaces, the models have been adopted widely.

Large language models have also been used for information extraction. An LLM with a zero-shot approach can even extract previously unseen attribute values [9]. It is also suitable for tasks that require recognizing multiple attributes and their relationships simultaneously [10]. LLM can also be used to normalize attribute values [11].

The challenge with the LLMs is that a poorly formulated prompt can lead to poor results. It is also strongly a black-box approach, with internal functions not transparent, making it difficult to trace bugs. Zero-shot, few-shot, and chain-of-thinking (CoT) approaches were compared for the extraction of character properties from movie manuscripts in [12]. Zero-shot worked well for extracting clear properties, but accuracy improved significantly when a few examples (few shots) were supplied. In their results, CoT achieved the best accuracy for complex and implicit properties, but at the cost of increased false negatives.

Raj et al. [13] studied the consistency of the extraction results and concluded that it is a central factor in assessing accuracy. The more consistently the model produces similar results for semantically similar inputs, the more reliable it can be considered. They aimed to strengthen consistency by fine-tuning the LLM.

In this paper, we take a similar approach to [13] but with the opposite viewpoint. Our goal is not to make the models more consistent, but to evaluate their existing consistency to improve accuracy. We achieve this by repeating the extraction process and using consistency as a measure of the model's reliability for a given extraction result. If the model was 100% consistent (always returning the same result), it would lose diversity and be unable to assess the confidence of the extraction result.

So, from this viewpoint, small variations are not considered a problem but a signal that helps evaluate the reliability of the extraction result. For example, if the product information contains two contradictory values for the same attribute value, a fully consistent LLM would just pick one of them and appear reliable, even if the source data has factual uncertainty. Therefore, analysis based on

repeated extractions shows these kinds of contradictions as variation, which reveals the sample is ambiguous.

The idea that consistency of LLM results correlates with reliability and uncertainty is not entirely new. Wang et al. introduced a self-consistency method [14], in which an LLM generates multiple reasoning paths**,** and the most consistent answer is chosen as the result. Cheng et al. [15] analyzed the consistency of multiple samples to conclude the certainty of a single sample. Rivera et al. [16] analyzed the mutual consistency of the response as a tool to estimate the uncertainty of an LLM and as a signal of the unreliability of LLM-generated text. Our work extends these studies by applying consistency as a measure of reliability in attribute-value extraction.

3. Consistency-enhanced LLM-based attribute value extraction

Next, we present our variation of the LLM, called **C**onsistency-**E**nhanced **LL**M-based **A**ttribute **V**alue **E**xtraction (*CELLAVE*). The method is relatively simple and consists of the following four steps:

1) **Extraction**: Repeat LLM multiple times for the attribute-value extraction.
2) **Selection**: Select the most frequent output as the candidate result.
3) **Consistency**: Calculate how many times (%) the selected result was obtained
4) **Acceptance**: Accept the result if the consistency is higher than *T*.

We used prompts derived directly from the meta-information of every possible attribute without any hand-crafting.

For the LLM component, we used the GPT-4o model (version 2024-08-06), which was state-of-the-art at the time of the experiments. At the time of writing this paper, OpenAI GPT-5.5 and Anthropic Claude Opus 4.7 might be more popular already, and when you read this paper, it might be something completely different. This is how fast the LLM methodology evolves nowadays. The parameters used for GPT4o were the following: temperature = 1, top-p = 1.0, n = 5, max tokens = 2000.

We repeat each extraction five times and select the most frequent output as the result. We define consistency as the percentage of the extractions that matched the most common extraction. For example, if we extracted the same color four out of five times, the consistency is 80%, see Figure 3.

The result is accepted if the consistency exceeds a threshold. With five repeats, we can have five possible thresholds (*T*): 20%, 40%, 60%, 80%, 100%. For example, with *T*=60%, the extraction result is accepted only if the same value was extracted at least 3 times. Additional repeats can provide a more fine-tuned consistency measure, but the 20% step size was found sufficient for our experiments.

Figure 3. Calculating consistency from repeated extractions.

Prompt design:

The input parameters to the GPT-4o model included instructions, product information, the attributes to be extracted, and the output format (JSON). We used the expert role of product

information manager for LLM. To minimize hallucination, we explicitly instructed not to invent new information and to use merely the given data. The prompt we used is shown in Figure 4.

- → You are a product owner with 20 years of experience. You are tasked with selecting attribute values for products.
- → "Product data" the data about the product in question.
- → "Attributes data" contain attributes that need to be extracted. Use "attribute_code" to refer to the extracted attribute.
- → If applicable, use proper unit for each extracted attribute based on the attribute "dimension" by using the "unit_code".
- → Only use the given data and do not make up values. Use null when the value cannot be found in the data and prefer not to return those at all.
- → Only respond using the "Response model" json.

Figure 4. Prompt used in our experiments.

The product information was input in all available languages. An example is below:

- Name: “Sikaflex 221 Strong adhesive and sealing compound, black, 300 ml”
- Description: “Sikaflex®-221 is a high-quality, multipurpose, non-sag one-component polyurethane adhesive and sealant that…”

(These were originally in Finnish, and we translated them into English)

The attributes and their descriptions were entered in a structured form, including the unit codes used, which allowed the attribute value in different units to be standardized for later conversions. For example:

- Attribute name: “*Volume*”
- Description: “*The volume of liquid contained in the product*”
- Data type: Numeric, integer, text, boolean, or value from a selection list.
- Accepted units and their information: Liter (code: L), deciliter (code: DL), milliliter (code: ML)
- In the case of a selection list, the accepted values: Summer tire, studded tire, friction tire.

For example, if the expression “*300 ml*” appears in the product item name and the attribute to be extracted is *volume*, the correct extraction result would be 300 and unit code ML.

4. Experimental setup

We compiled a dataset from a real product information system provided courtesy of the Broman group. The database contains a wider range of products from lamps to car tires. Information stored includes product title, product description, and metadata with existing attribute-value pairs. Most products have descriptions in three main languages: Finnish, Swedish, and Estonian. Some products also have information in English and Russian.

There are 31,121 products in total, divided into 189 categories. There were 40 different attributes to be extracted with varying structures. There were attributes of the following types: numerical (such as weight in grams), Boolean (such as whether a helmet includes a sun visor), and categorical (such as the noise class of a tire). The text description served as the source data fed to the LLM for information extraction, and the existing attribute-value pairs served as ground truth to evaluate how well the extraction succeeded.

The attributes and their frequencies are summarized in Table 1. The data contained a particularly large number of car tires, leading to their attributes being overrepresented. For instance, car tires have attributes such as rim size that are very easy to extract. For this reason, we use the macro-average

(MA) of the proportion of correct and incorrect extractions so that all attributes contribute equally to the evaluation.

Table 1. The data contained 40 different attributes in total.

Attributes	Values
Speed rating	21 859
Tire width (mm)	21 567
Tire profile	21 423
Rim size	21 049
Load index	19 381
Tire type	18 291
Wet grip	17 662
Fuel efficiency	17 629
Noise class	16 011
Noise (dB)	15 806
Color	3 053
Size	2 118
Weight	1 296
Lock type	774
Multiple shell sizes	774
Sun visor	742
Places for helmet phone	742
Pinlock ready	742
Pinlock included	742
MIPS	730
Material	660
Lamp type	526
Power	502
Bulb base	490
Shade type	481
Dimmable	481
Color temperature	478
Luminous flux	473
Operating voltage	471
Color temperature (K)	441
Length	436
Caliber	235
Height with dimension	231
Cable length (m)	162
Removable interior	146
Impact strength	60
Number of main bulbs	50
Bicycle tire size (inches)	23
Tire width (inches)	23
Bicycle tire size (mm)	23

5. Results

The dataset contains 208,783 product-attribute pairs. LLM extracted at least one value for 139,858 of them (67%), while the remaining 68,925 (33%) remained empty. In the evaluation, we divided the dataset into two parts: (Set A) product-attribute pairs explicitly present in the data (138,902 pairs); (Set B) product-attribute pairs absent from the data. (69,881 pairs) The extractions are divided into the following categories:

Set A: Product information exists (138,902 pairs)

- True positive (TP): successful extractions with correct information.
- False positive (FP): extraction with incorrect information.
- False negative (FN): LLM failed to extract.

Set B: Product information is missing (69,881 pairs)

- True negative (TN): LLM did not hallucinate the information.
- False positive (FP): LLM hallucinated information.

TP and FN appear only in set A, and TN only in set B. False positives (FP) can appear in both with different interpretations. In set A, FP means incorrect extraction. In set B, FP means hallucination. Set B was tailored to assess hallucinations exactly. To assess the results, we use three measures:

$$Precision = \frac{TP}{(TP+FP)}, \tag{1}$$

$$Recall = \frac{TP}{(TP+FN)}, \tag{2}$$

$$Specificity = \frac{TN}{(TN+FP)}. \tag{3}$$

Precision and recall are measured from set A, and specificity from set B. Precision assesses the correctness of the extractions: the less incorrect information, the higher the precision. Recall assesses coverage: the fewer values we miss, the higher the recall. Specificity assesses the amount of hallucination: the fewer the extraction results that do not exist in the data (false positives), the higher the specificity.

Each product attribute was extracted five times. The analysis was carried out in two different ways. (1) Every five extractions are treated as an independent observation corresponding to a baseline LLM. (2) The most frequent result is selected (consistency-enhanced extraction). The consistency percentage is used to decide whether to accept or reject the extracted value.

The lowest threshold (20%) means that the result is the most frequent extraction regardless of consistency. The highest threshold (100%) means that only results with full agreement by all five extractions are accepted. Rejecting less consistent extractions has the potential to improve precision. Fully consistent extractions account for 91.6% of all cases. With the strictest (100%) threshold, 8.4% of the extractions would be rejected, see Table 2. The lower threshold (80%) would accept 96.9% of the extractions with 3.1% rejections.

The main results in Figure 5 show that the precision is high: 99.5% of the extracted values are correct. However, it missed 2.6% of the attributes (97.4% recall). Repeating the extraction with the lowest 20% threshold improved precision from 99.5% to 99.6%, but more importantly, it increased recall from 97.4% to 99.3%. In other words, we can find more results simply by repeating the extractions. Specificity decreased slightly from 99.1% to 98.9% due to a few more hallucinations.

The effect of the threshold is analyzed in Figure 6 by varying consistency from 20% to 100%. The highest 100% threshold requires that all extractions provide the same result. The precision further

increased from 99.6% to 99.8%, which means that incorrect extractions reduced from 0.4% to 0.2%. As a downside, recall dropped from 99.3% to 88.1%, meaning we missed 18 times as many attributes: 923 vs. 16629. However, the main benefit is that we got rid of hallucination completely (from 1538 to 0) as specificity increased from 98.9% to 100%.

Table 2. Amount of extracted attributes with different consistency values.

Consistency	Count	Percentage
20	700	0.5 %
40	1,317	0.9 %
60	2,340	1.7 %
80	7,425	5.3 %
100	128,076	91.6 %
Total	**139,858**	**100%**

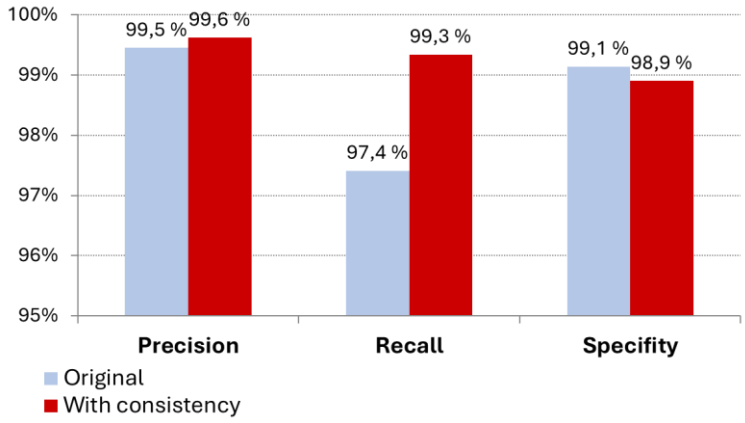
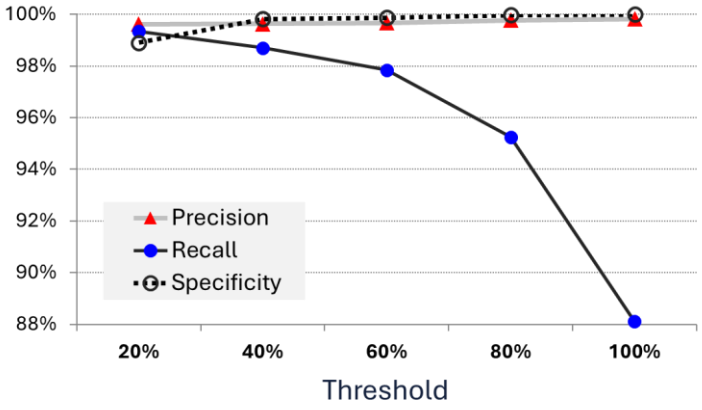


Figure 5. Overall extraction results (macro-averages).



Values:

T	Precision	Recall	Specificity
20%	99.6%	99.3%	98.9%
40%	99.6%	98.7%	99.8%
60%	99.7%	97.9%	99.9%
80%	99.8%	95.3%	100%
100%	99.8%	88.1%	100%

Figure 6. Effect of the threshold on the measures (macro-averages).

6. Cost estimation

To put the results in a better perspective, we relate them to hypothetical human work. We have three different strategies:

1) Totally human work;
2) Totally AI work;

3) AI-assisted human work.

In terms of cost estimation, we consider only Set A, because values for Set B must be obtained from completely different source. In the AI-assisted strategy, we use the highest consistency threshold to get rid of hallucinations. The missing values are left to human work. With the assumption that filling these missing values takes the same time as filling any other value, we have only 16,629 values to check, compared to 138,902 values if humans extracted all values, see Table 3.

Table 3. Amount of human work and errors in three different strategies.

	Human work		Errors		
Strategy	**Attributes**	**Hours**	**Incorrect**	**Hallucinations**	**Total**
1) Human work	138,902	32.2 days	972	0	972 (0.7%)
2) Baseline LLM	3,636	20.2 h	699	1259	1958 (1.4%)
3) LLM (T=20%)	923	5.1 h	531	1538	2069 (1.5%)
4) LLM-assisted (T=100%)	16,629	3.8 days	238	0	238 (0.2%)

6.1. Human work

Let us assume that humans would process one attribute in about 20 seconds. This is highly hypothetical, but a concrete number can make the comparison a bit easier. Filling in 138,902 attributes would then require 32.2 days' nonstop work. Using an LLM, we can reduce the human workload to less than 5 hours, consisting of filling in the values that the LLM failed to extract. The downside is 531 erroneous values (0.4%) and 1538 hallucinated values (1.1%). Using a consistency-enhanced LLM, we can reduce erroneous values to 238 (0.2%) without any hallucinations (0%), but at the cost of more human work (about 4 days).

In other words, using CELLAVE with maximum consistency, we can reduce the total errors from 2069 (531 incorrect detections + 1538 hallucinated values) to 238 errors at the cost of increasing human work from 5 hours to 4 days (nonstop). This may sound costly, but it is still remarkably less than if all attributes were extracted by a human, which would require about 32.2 days.

This analysis also assumes that humans make no errors, whereas LLM has an error rate of 0.2%. In practice, human errors also happen. Furthermore, when an LLM does not provide a consistent result, it can help humans pay attention. The LLM's extraction can also be used as an initial suggestion that humans need to verify.

To sum up, using the proposed consistency-enhanced LLM, we can reduce human work to 12% at the cost of a 0.2% error rate. This means 238 erroneous values in about 140,000 attribute values. We can compare this to our estimate of the human error rate as follows. We counted the number of attribute-value pairs with contradicting information values in the database compared with those in the product information. We found 1522 pairs, corresponding to an error rate of 0.7%.

6.2. Costs

The data extraction of 208,783 attributes-value pairs using the Azure OpenAI API cost approximately 800€, about 0.4 cents per extraction (including repetitions). The pricing of OpenAI's language models is based on both input and output length, but when the same input is used multiple times in a single query, only the cost of the outputs scales with the number of repetitions. In this case, the cost of extraction, including repetitions, can be expressed as:

$$\text{Cost} = \text{Cost(input)} + n \cdot \text{Cost(output)}, \tag{4}$$

where *n* is the number of repetitions. The cost of the input dominates, as the number of characters in the inputs is significantly larger than in the LLM-generated outputs. Thus, the repetitions do not significantly increase expenses.

By human work, we estimated 32.2 days nonstop. With 21 working days per month and 7.5 hours per day, the extraction takes about 5 months' work. According to [17], the median gross salary of a full-time wage earner in Finland in 2023 was 3,289€/month. If an employee performed data extraction continuously, the work of five months would cost over 16k€ gross salary plus overheads. This exceeds the cost of AI work by 20-fold.

The LLM-assisted strategy would cost 800€ + roughly 2 weeks' human work, totaling 800€ + 1900€ = 3700€. In other words, the three strategies lead to the following costs:
(1) Human work: 18k€ (0.7% incorrect values);
(2) Standalone LLM:<1k€ (1.5% incorrect values and hallucinations);
(3) LLM-assisted: 3.7k€ (0.2% incorrect values).

In addition, set B must be solved by a completely different strategy using different source.

6.3. Risks

Incorrect size can cause problems in storage, picking, transportation, and handling returns. Product returns add logistical burden and decrease the customer experience. Although such errors do not necessarily pose an immediate safety risk, their business significance can be substantial. On the other hand, not all errors are equally critical. Errors that reduce product discoverability are more visible than inaccuracies in less important attributes, such as a lamp's color temperature or the type of shade.

The most serious risk concerns situations in which incorrect or missing product information may harm health or safety. For example, missing allergen information can lead a customer to mistakenly believe the product is safe to use when it is not. In such cases, an information extraction error is not just a quality issue; it can have serious consequences.

A good strategy might be to limit automatic extraction only to low-risk attributes or at least require human verification of all high-risk attributes. Human inspection can also focus on non-consistent extractions.

7. Conclusions

We have introduced a consistency-enhanced LLM for product-attribute extraction. The idea is to repeat the extraction multiple times, select the most frequent extraction, and reject most inconsistent extractions. The proposed variant reduces error rate from 0.5% to 0.2% and, more importantly, eliminates all hallucinations (specificity increases from 99.1% to 100%). When used in AI-assisted human attribute extraction, it requires only a fraction of the time humans do. Compared to human error rate of 0.7%, the accuracy can be even improved. However, remember that the information is available only in part (138,902 out of 208,783) of the product documents. The rest still depend on human work to collect information elsewhere.

Data availability

The data consists of the company’s product information, which is commercially significant. For this reason, the data is not publicly available. Consequently, we do not present raw data or internal

data structures. Collaboration with the company demonstrates practical relevance but also sets a limitation: others cannot replicate the results using the exact same data.

Use of AI tools declaration

GPT-4o was used to extract product features, which is integral to the research question. AI was not used to conduct the analysis, calculate results, or implement the research methods. The research design, data processing, analysis, interpretation of results, and conclusions were all developed by the authors. ChatGPT was used to improve the readability of an early draft version of the paper. AI-generated suggestions were evaluated critically, and no AI-generated text was copied directly into the paper. The authors bear full responsibility for the content and have followed good scientific practice.

Conflict of interest

Pasi Fränti is an Editor-in-Chief for Applied Computing and Intelligence and was not involved in the editorial review and the decision to publish this article. The authors declare no conflict of interest.

References

1. U. Asklund, A. Dahlqvist, *Implementing and integrating product data management and software configuration management*, 2003, Norwood: Artech House.
2. *Akeneo, 2023 B2C survey results report: what shoppers want*, Akeneo company, 2025. Available from: https://www.akeneo.com/white-paper/2023-b2c-survey-results-report/.
3. J. Abraham, *Product information management: theory and practice*, Cham: Springer, 2014. https://doi.org/10.1007/978-3-319-04885-7
4. G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, C. Dyer, Neural architectures for named entity recognition, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, 260–270. https://doi.org/10.18653/v1/N16-1030
5. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, et al., Attention is all you need, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, 5998–6008.
6. J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, 2019, 4171–4186. https://doi.org/10.18653/v1/N19-1423
7. Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, et al., RoBERTa: a robustly optimized BERT pretraining approach, arXiv: 1907.11692. https://doi.org/10.48550/arXiv.1907.11692
8. J. Kunz, Understandinglarge language models: towards rigorous and targeted interpretability using probing classifiers and self-rationalisation, Ph.D. Thesis, Linköping University, 2024.
9. J. Gong, H. Eldardiry, Multi-label zero-shot product attribute-value extraction, *Proceedings of the ACM Web Conference 2024*, 2024, 2259–2270. https://doi.org/10.1145/3589334.3645649
10. J. Dagdelen, A. Dunn, S. Lee, N. Walker, A. Rosen, G. Ceder, et al., Structured information extraction from scientific text with large language models, *Nat. Commun.*, **15** (2024), 1418. https://doi.org/10.1038/s41467-024-45563-x

11. A. Brinkmann, N. Baumann, C. Bizer, Using LLMs for the extraction and normalization of product attribute values, In: *Advances in databases and information systems*, Cham: Springer, 2024, 217–230. https://doi.org/10.1007/978-3-031-70626-4_15
12. S. Baruah, S. Narayanan, Character attribute extraction from movie scripts using LLMs, *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, 8270–8275. https://doi.org/10.1109/ICASSP48485.2024.10447353
13. H. Raj, V. Gupta, D. Rosati, S. Majumdar, Improving consistency in large language models through chain of guidance, arXiv: 2502.15924. https://doi.org/10.48550/arXiv.2502.15924
14. X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, et al., Self-consistency improves chain of thought reasoning in language models, arXiv: 2203.11171. https://doi.org/10.48550/arXiv.2203.11171
15. F. Cheng, V. Zouhar, S. Arora, M. Sachan, H. Strobelt, M. El-Assady, RELIC: investigating large language model responses using self-consistency, *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, 2024, 647. https://doi.org/10.1145/3613904.3641904
16. M. Rivera, J. Godbout, R. Rabbany, K. Pelrine, Combining confidence elicitation and sample-based methods for uncertainty quantification in misinformation mitigation, *Proceedings of the 1st Workshop on Uncertainty-Aware NLP (UncertaiNLP 2024)*, 2024, 114–126. https://doi.org/10.18653/v1/2024.uncertainlp-1.12
17. *Statistics Finland, Tulorekisterin palkat ja palkkiot (Finnish)*, Tilastokeskus, 2025. Available from: https://stat.fi/tup/kokeelliset-tilastot/tulorekisterin_palkat_ja_palkkiot/index.html.

AIMS Press