Data Completion in E-commerce

Liat Antwarg Friedman
eBay Research
lantwargfriedman@ebay.com

Gal Lavee
eBay Research
glavee@ebay.com

Bracha Shapira
eBay Research
bshapira@ebay.com
bshapira@bgu.ac.il

Dorin Shmaryahu
dorins@post.bgu.ac.il
Ben-Gurion University
Israel

ABSTRACT

In e-commerce, incomplete product data presents a significant challenge, particularly in online marketplaces where small businesses and individual sellers often lack the resources to provide full product details. Missing data in product items—whether structured fields like feature-value pairs or unstructured text such as titles and descriptions—can hinder product search, recommendations, and overall marketplace functionality. Addressing these data gaps is essential for enhancing the efficiency and user experience of e-commerce platforms.

This paper introduces low-cost machine learning approaches for completing missing textual features in such items, offering an alternative to computationally expensive large language models (LLMs). We propose two cost effective methods: the first extracts data from unstructured fields like item titles or descriptions, while the second employs a Nearest Neighbors method to impute missing values based on similar items. Both methods are evaluated on real-world datasets across diverse product categories, including sports trading cards, motor parts, and computers. Our experiments demonstrate that these low-cost approaches can achieve performance comparable to LLMs, often at a fraction of the computational cost, making them a viable option for large-scale e-commerce platforms. We also demonstrate that completing missing data not only improves data quality but also enhances key tasks such as search optimization and matching items to catalog products, which are critical for e-commerce platforms.

1 INTRODUCTION

Data plays a critical role in e-commerce, where vast amounts of product information, customer data, and transaction records must be organized for efficient processing [1]. Structuring this data into constructs such as tables allows efficient queries on various product features such as price, category, brand and color, e-commerce platforms can facilitate faster product retrieval, personalized recommendations, and better inventory management. This organized structure reduces duplication, improves accuracy in customer orders, supports data-driven decision-making, ultimately contributing to business growth and customer satisfaction [2, 3]. When organizations supply data, it is typically structured and follows predefined schemas, simplifying processing, analysis, and integration. Enforcing structured data is feasible when a few players generate large datasets through automated processes, adhering to strict protocols.

However, there are applications where many small pieces of data are created and uploaded by numerous small players, who may not have the motivation or capabilities to provide all the needed meta data. Consider for example the case of an online marketplace, such as eBay[1], where small businesses and individuals sell items. An individual is expected to sell only a handful of items, and small businesses do not have the resources to develop automated processes for uploading new items. The marketplace, on the other hand, desires to lower the burden on sellers that upload items, in order to avoid discouraging them from using the platform due to significant efforts for selling low profit items. Hence, many sellers in such online marketplaces often upload partial information about the item that they sell.

In this paper we focus on the problem of processing items that sellers upload in such online marketplaces, attempting to fill in some of the missing data that the seller did not provide. Completing missing data in e-commerce is essential for ensuring the efficient operation of online marketplaces, as it directly impacts critical processes such as search, recommendations, and catalog management. Addressing these data gaps enables more effective search optimization, catalog matching, personalized recommendations, etc. . In this work, we demonstrate the importance of data completion by focusing on the use case of catalog matching, a fundamental task for linking seller-provided item listings with structured catalog entries, thereby improving search accuracy and user experience.

The listed item is the core unit of an online marketplace. It is a particular physical instance of product that a seller offers for sale on the marketplace. Each item is multi-modal, comprised of, e.g., product images, a title (short unstructured text), a description (long unstructured text), and feature-value pairs (structured text data).

We suggest two low-cost approaches to complete empty features values thus increasing the quality of the uploaded items, focusing only on textual features. Several papers have suggested using large language models (LLMs) for completing missing data [5, 7]. Such models encompass, in a way, all the data that exists online, and can hence already have

[1] www.ebay.com

encoded in their structure all the missing information. However, the computational effort required for using such models is huge, which may translate to a significant financial cost for the online marketplace. We therefore focus on low cost machine learning approaches for data completion, comparing them to the quality of data completion of an LLM. We suggest 2 approaches, and within each approach implement a wide set of variants.

The first approach is based on data extraction [11, 13, 22]. In many cases sellers upload the required information in an unstructured manner, in the item title or description, but not in the designated structured fields. This approach analyzes the unstructured data in an effort to identify and extract the missing values. We create a statistical model based on a large set of items, that associates terms from the unstructured data provided by sellers to features. Variants of this approach contain extracting data from either the title or item description.

The second approach is based on similar items. In a huge marketplace such as eBay instances of a given product are often sold by many sellers. One can use the information in identical products, or in similar products, to complete the missing values [17]. The challenge is to properly identify the most similar products based on the given partial information. We suggest a Nearest Neighbors approach that uses embedding — a numeric vector representation of an item. We use a pre-trained text model, such as CLIP [14], to create an embedding for an item based on its unstructured content. For a new item, we compute its embedding and find a set of Nearest Neighbors among the previous items in the embedding space. We then use the feature values of the neighbors to complete the missing values in this new item. Variants of this approach include computing the embedding either in the title of the item or on the item description. Another variant is to fill the missing values with the most dominant values from the closest neighbors.

We provide a set of experiments that compare the methods from the above approaches and an LLM-based data imputation method. We used 3 datasets from different diverse categories – sports trading cards, motor parts, and computers. We collected in each category almost 1 million real items uploaded by sellers. We ran a wide set of variants from each approach on the datasets and compare their performance. Our results show that while LLMs provide strong performance, in many cases our methods produce similar, and sometimes better results, with a fraction of the needed cost. This provides a strong incentive for organizations to prefer low-cost approaches in their production systems.

The contribution of this paper is the introduction of a novel, low-cost machine learning approach for completing missing feature values in e-commerce, utilizing data extraction and nearest-neighbor techniques as scalable and efficient alternatives to large language models (LLMs). While embedding-based similarity is a well-established concept, its application to structured feature completion in e-commerce listings is largely unexplored. Our approach demonstrates performance comparable to LLMs while significantly reducing computational costs, offering a practical and accessible solution for large-scale e-commerce platforms. From an industry perspective, we demonstrate that leveraging embedding similarity for missing data completion improves candidate retrieval—the critical first phase in matching items to catalog products. This advancement directly enhances search relevance, optimizing product discovery and the overall user experience in e-commerce buying sessions.

2 RELATED WORK

In e-commerce, noisy data is a common issue stemming from incomplete inputs, inconsistent descriptions, and diverse customer behaviors [1, 8]. Approaches to address missing data range from basic statistical imputation to advanced machine learning. Simple methods, such as mean, median, or mode imputation, are computationally efficient for large datasets but often introduce bias, underestimate variability, and fail to capture complex relationships between variables [10]. These methods are particularly limited for non-numeric data, which is prevalent in e-commerce. Advanced methods, like Multiple Imputation by Chained Equations (MICE) [19], model each missing feature through sequential regressions. While effective for structured data, MICE struggles with text data due to its reliance on simpler predictive models. Similarly, K-Nearest Neighbors (KNN) imputation leverages similarities between records but becomes computationally expensive with large datasets [17].

Hameed and Ali [6] compared imputation techniques, finding that mean imputation underperforms, while KNN offers moderate improvements. MICE outperformed both, but faced scalability and execution time challenges with large datasets. Recently, machine learning models have gained traction for data imputation. Unlike traditional methods, machine learning models can capture complex relationships between features, making them more effective in datasets with non-random missing patterns. Random Forest Imputation, introduced by Stekhoven and Bühlmann [15], uses an ensemble of decision trees to predict missing values based on the observed data. Stekhoven and Bühlmann [15] employs decision tree ensembles to predict missing values, offering robust handling of complex feature interactions and improved scalability for large datasets.

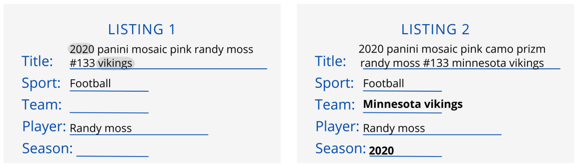
Deep learning models have also been explored for imputation. Generative Adversarial Networks (GANs), have been used to generate synthetic data to fill missing values [21]. GAN-based approaches have demonstrated superior performance, particularly in datasets with complex feature interactions, offering a compelling alternative to traditional imputation methods. We decided not to use GANs due to their complexity and limited scalability in industrial settings. Large Language Models (LLMs) have shown promise as generative tools for imputing missing data in various fields [9, 14, 18]. In e-commerce, LLMs are emerging as tools to fill missing product attributes, such as descriptions or specifications, by generating contextually relevant text. Their ability to produce semantically consistent data provides a significant advantage over traditional methods, although high computational costs remain a challenge [5, 7]. Wang et al. [20] compared traditional methods such as MICE and CART with modern deep learning approaches like GAIN and MIDA, finding that while deep learning models offered faster computation, MICE-CART consistently outperformed in terms of bias, mean squared error, and coverage. Similarly, Sun et al. [16] evaluated these methods across various data types, missing mechanisms, and correlation levels, revealing that conventional methods still excel in scenarios with limited sample sizes. Lastly, Osman et al. [12] proposed a "top-down bottom-up" approach to select imputation techniques, combining analysis with selection to suit specific domains.

3 FEATURE VALUE COMPLETION APPROACHES

In this section, we outline our proposed methods for filling missing product feature values. Our aim is to eliminate reliance on costly human-labeled data, enabling low-cost scalability across diverse e-commerce categories. No labeled data was used in the filling process; it was used only for evaluation.

Problem definition: An *item* describes an e-commerce item for purchase. A *category* C is a set of items that are related in some sense. Each item l belongs to a single category $C(l)$. Items in a category are described by a set of features $F(C)$, categorical, numeric, or textual. That is, each item $l \in C$ is described by assigning a value to features in $F(C)$. A complete assignment of values to the features of a category uniquely identifies an item. That is, given two items with identical feature values, buyers should be indifferent as to whether they receive any of them.

We denote by l_f the value that item l assigns to feature f. The value for l_f may be empty (missing), that is, there is a correct value for l_f, but that value is unknown. We differentiate here between a *null* value, denoting that the item should not have a value for a specific feature, and a *missing* value. The goal is to complete such missing values, i.e., compute a value $\hat{l}_f$, such that $l_f \equiv \hat{l}_f$, that is, $\hat{l}_f$ may not be identical to l_f, but it has equivalent meaning. We focus on completing missing values only for categorical and numerical features, excluding textual data like product descriptions.



LISTING 1

Title:	2020 panini mosaic pink randy moss #133 vikings
Sport:	Football
Team:	
Player:	Randy moss
Season:	

(a)

LISTING 2

Title:	2020 panini mosaic pink camo prizm randy moss #133 minnesota vikings
Sport:	Football
Team:	**Minnesota vikings**
Player:	Randy moss
Season:	**2020**

(b)

Figure 1: Two items with similar titles. Goal: complete the values in features 'team' and 'season' in (a).

3.1 Extracting Data from Free Text

We now present a set of unsupervised learning methods for extracting feature values from free text available in the item. Such free text may exist in the item features such as the title, or the description. Indeed, in many cases sellers attempt to make the item title as descriptive as possible, adding many feature values into it, such as the item brand, model, color or even more specific values such as shoe size.

The methods suggested here utilize the information available in the items, without incorporating any external data. Our methods contain an offline computation phase, where we pre-compute a model that is later used online, as new items are uploaded, to complete the missing feature values.

Offline phase. For this family of methods, we compute offline a probability distribution $pr(f|v, C)$ where v is a value, C is a category, and f is a feature. That is, given a value v extracted from some textual description, we can use $pr(f|v, C)$ to identify to which feature it should belong. This method is category dependent, as features and values can have different meaning in different categories.

Value-Feature Probability Distribution: The probability distribution is computed over a large set of items. In the set, each feature contains values that were uploaded by the sellers. To assign a value to a feature, we first compute for each feature the relative frequency for each value in the items in a given category. Then, after computing for each value its relative frequency within a feature, we normalize to obtain $pr(f|v, C)$. Here, our methods use only the maximal feature: $f_{max}(v) = \arg\max_f pr(f|v, C)$.

That is, for each value v we map it only to its most likely feature $f_{max}(v)$. However, one can envision extensions that leverage the probability distribution. For example, if the most likely feature already has a value, and it is different than v, we can consider the second most likely feature, and so forth. we leave such extensions to future research.

Terms Extraction: To identify potential feature values in the textual item features, we identify terms in the unstructured text in the items (e.g., title, description) using the Phrases method from the Gensim[2] package. The Phrases method in Gensim is based on identifying collocations or frequently co-occurring words in a corpus. The method primarily extracts bigrams (two-word phrases) or n-grams (n-word phrases) that are more meaningful when considered together than as individual words. (e.g., "New York" or "machine learning"). Gensim's Phrases method focuses on finding these collocations by analyzing the co-occurrence statistics of words in a text corpus. In our experiments we report separately the extraction of terms from (1) the title, and (2) the description. In the online phase we only consider terms that were identified in the offline phase.

Online phase. Given an item l, we extract terms from its textual features, using the list of terms from the offline phase. For each such potential value v, we find its most likely feature $f_{max}(v)$ calculated during the offline phase. If the value of $f_{max}(v)$ is missing, we assign v to $f_{max}(v)$.

There are two variations to this approach: the extraction of terms can be from (1) the title, (2) the description, that is, the distribution $pr(f|v, C)$ is based only on terms extracted from titles or descriptions during the offline and the online phases. Hence, this data extraction approach creates four different variations. Figure 1a presents an example of an item with a title and structured features, where two of its features (Team and Season) have no values. In order to complete these features values using data extraction, the following terms are extracted from the title: '2020', 'panini', 'randy

[2]https://pypi.org/project/gensim/

moss', and 'vikings'. Using the Value-Feature probability distribution created in the offline phase, the term '2020' is associated with the feature Season and the term 'vikings' is associated with the feature 'Team'.

Limitations. A primary limitation of this approach arises when a term from the test items does not exist in the pre-computed model making it impossible to infer the most likely feature it belongs to.

3.2 Nearest Neighbors

Offline Phase. We compute and store an embedding for a representing text of each item in the unlabeled dataset. These embeddings are generated based on pre-trained models and represent each item in a vector space. We embed the available representing text of the title or the description.

The pre-trained models we used in the experiments in this paper are : CLIP Radford et al. [14] and two other models based on BERT [4] that were fine-tuned on eBay's data.

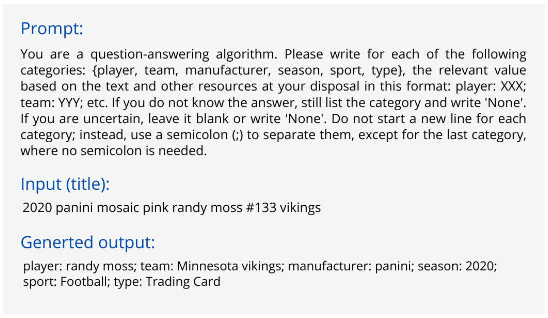
Online Phase. When a new item is encountered, we compute the embedding of the representing text (title or item description) using the same pre-trained models. This embedding is then used to find the set of Nearest Neighbors from the embeddings calculated during the offline phase. The Nearest Neighbors are determined using the cosine similarity of embeddings, under the assumption that similar items possess similar feature values. For each missing feature value in the item, we assign a value derived from the Nearest Neighbors using one of the following two variations: the most similar nearest neighbor (above a similarity threshold), or the majority from the nearest k neighbors. This approach is justified by the assumption that the existing data in the database have already undergone a refinement or improvement process, making the neighborhood feature values reliable indicators. Assume that the item in Figure 1b is the closest match to the item in 1a based on their embedded titles. From the item in Figure 1b, we take the missing feature values for 1a.

Limitations. As with any embedding-based method, the accuracy of feature values prediction is contingent on the quality of the pre-trained models and the assumption that similar products, as measured by their embeddings, share similar feature values. Thus, the method will not work well in cases where the test item has no sufficiently similar counterparts in the dataset, or if the embeddings fail to capture key semantic differences.

3.3 Large Language Model (LLM)-Based feature values Generation

This method leverages a pre-trained Large Language Model (LLM), such as GPT-4 or GPT-3.5-turbo, to generate feature values for all items in the test set based on input text (e.g., titles, item descriptions), in a zero-shot form.

However, in future work, we can explore the use of few-shot inference, where a small number of examples per feature are provided to the LLM. This approach could further improve the accuracy and relevance of the generated feature values by offering the model more task-specific context and guidance. Such enhancements may also help the model better handle domain-specific variations. While this exploration of LLM improvements is reserved for future work, our primary focus here was to assess the value of using simple, low-cost methods in comparison to LLMs. These methods are often more efficient, easier to implement, and easier to scale in production environments and, in many cases, provide a good-enough solution that significantly improves upon the current situation, even if it does not achieve the best possible outcome. This approach has only an online phase (unlike the previous two approaches). Using a custom-designed prompt, we instruct the model to predict relevant features' values for each item. Figure 2 is an example of the prompt used for the *sports* category, and an input-output example. The input title is equal to the title in Figure 1a. In this example, the model's output includes details not explicitly stated in the input title, such as the sport and type, inferred from context and internal knowledge.


Prompt:

You are a question-answering algorithm. Please write for each of the following categories: {player, team, manufacturer, season, sport, type}, the relevant value based on the text and other resources at your disposal in this format: player: XXX; team: YYY; etc. If you do not know the answer, still list the category and write 'None'. If you are uncertain, leave it blank or write 'None'. Do not start a new line for each category; instead, use a semicolon (;) to separate them, except for the last category, where no semicolon is needed.

Input (title):

2020 panini mosaic pink randy moss #133 vikings

Generted output:

player: randy moss; team: Minnesota vikings; manufacturer: panini; season: 2020; sport: Football; type: Trading Card

Figure 2: Prompt for generating sport related features

Limitations. Running LLMs at scale can be resource-intensive in terms of computation and memory, particularly for large product catalogs with many missing values. The cost of generating values with LLMs can be prohibitive for large-scale applications.

4 EXPERIMENTS

4.1 Datasets

In order to evaluate our approaches to complete missing feature values in e-commerce items we collected data from *eBay*, one of the world's largest e-commerce platforms. The data was collected across three high-level vertical categories of e-commerce.

These categories form three different datasets. Each of these high-level categories contains an assortment of product types. The first category, *Sports*, contains items such as trading cards and other collectibles as well as sports related apparel such as team jerseys. The second category *Motors*, contains items for automobiles and automobile components. The third category, *Computers*, contains items for personal computers, computer parts, and other computer related items. For each of the above categories we identified a small set of domain-specific high-priority features that we include in our evaluation. These features were chosen by domain experts based on their prevalence in the category and their importance to the buyer. For example, in the *Sports* category, many items, most notably collectible trading cards are associated with a particulate *Player* and *Team* as well as a particular *Sport.* If any of these features are missing or incorrectly specified the item may be harder to find or less appealing to a potential buyer. Table 2 enumerates the features associated with each category along with relevant information about their values, the number of unique values for each feature and their missing information ratio.

4.2 Data Collection

For each of the focus categories described above we collected both labeled (test set) and unlabeled data. The labeled data consists of a set of items for which the correct values for high-priority features have been manually annotated by trained human annotators. These high-priority features were chosen based on frequently searched features within a category and on domain knowledge. The annotators were given guidelines that they should write the feature values (can be more than one correct value. For example, a shirt can have more than one color) and in which signal (title, image, etc.) the value was found. For example, if the item is for a baseball card, the annotator should provide the team name as "New York Yankees" rather than "Yankees" or "NY Yankees". In addition, Annotators were instructed to provide all values that exist for each feature in the item they were annotating, as multiple values may be associated with a single feature. We used the labeled dataset as our test-set.

The unlabeled dataset contains a much larger set of items for which the correct values for the high-priority features are not known. The unlabeled dataset was used by our methods to assign values to missing data.

For both labeled and unlabeled datasets we collected all seller specified signals including the item title, description, and seller specified feature values. For each category, we used the unlabeled dataset for predicting values for items with missing values and the labeled dataset for testing our predicted value against ground truth, ensuring a comprehensive evaluation across different product categories. Table 1 shows the dataset statistics for *Motors*, *Computers*, and *sports*.

Table 1: Dataset Statistics for Motors, Computers, and Sports

Domain	Split	Total Listings
motors	Unlabeled data	846,340
motors	Labeled data	3,241
sports	Unlabeled data	885,641
sports	Labeled data	6,305
computers	Unlabeled data	997,474
computers	Labeled data	12,919

Sports Dataset. The *sports* dataset consists of 885,641 items in the unlabeled data set. The test set contains 6,305 items. The high-priority features that were chosen in this category are: *Manufacturer, Player, Season, Sport, Team, Type* where Type is the type of the item. For example, Sports trading card, shirt, hat, etc. In this category there are features with 11-36% missing values in the test set.

Motors Dataset. The *Motors* dataset includes product items related to the automotive category. The unlabeled set consists of 846,340 items, and the test set, significantly smaller, contains 3,241 items. The high-priority features that were chosen in this category are: *Compatible make, Compatible model, Compatible year, make, type.* The features 'compatible make', 'compatible model', or 'compatible year' specify the make, models and years that the item is compatible with. For example, if a motor works with multiple car makes, models, or production years, all relevant makes, models, or years will be listed under these features. In this category there are features with 31-97% missing values in the test set.

Computers Dataset. This dataset comprises items of various computer-related products. The unlabeled set includes 997,474 items, and the test set has 12,919 items.The high-priority features that were chosen in this category are: *Brand, Color, Model, MPN (manufacturer part number), and Type* - which indicates what type (i.e. computer part) of item is it - such as : motherboard, monitor, mouse, etc.. 'Compatible model', as in the *Motors* category have a very high percentage of missing values 88.4%. The other features have between almost no missing values to 48%. Some of the features have a lot of distinct values. The diversity of categories and the size of our datasets allow for robust testing of our methods, ensuring that the proposed techniques are applicable across various industries and product categories.

4.3 Metrics

We evaluate our methods on items with missing feature values from labeled datasets, where human annotators provide the correct values. Each feature is evaluated separately within its category, and for items with multiple missing features, each prediction is assessed individually as a success or failure. Since features may have multiple correct values, a prediction is considered correct if at least one predicted value matches an entry in the labeled list. During evaluation, for each missing feature, we generate a list of predicted values, $\hat{l}_f$, using each approach and compare it to the ground truth list, l_f. A prediction is correct if at least one predicted value appears in the labeled list. To measure the performance of the different approaches, we use the following list-based metrics: Precision measures how accurately models can assign the correct values, while Recall reflects how many relevant values the models can retrieve. There is often a tradeoff between these two metrics, but the best-performing models tend to balance them. Coverage measures the amount of retrieved values (without examining if they are correct). Furthermore, in our experiments precision and recall follow a "loose" definition. In this case, a prediction is considered correct if one of the predicted values is a subset of a label, or vice versa. For example, if the true label is "trading cards" and the predicted value is "sports trading cards," it would be counted as correct under the loose metric definition. Another metric is the F-score, calculated based on loose precision and recall. From all the variations we have for each approach, we report the best variation for each feature in each approach according to the highest F-score.

Table 2: Metadata about the categories, including main features, value example, number of unique values, percentage of missing features values in the unlabeled and test sets.

Domain	Feature	Value example	Unique feature values test	Unique values relative to test size	Missing feature values unlabeled data (%)	Missing feature values test (%)
Sports	Manufacturer	Topps	225	0.036	63.95	30.26
	Player	Michael Jordan	2360	0.374	64.03	27.58
	Season	2020	155	0.025	65.7	30.90
	Sport	Baseball	87	0.014	51.96	11.74
	Team	New York Yankees	622	0.099	63.08	36.15
	Type	Sport trading card	24	0.004	57.73	19.59
Motors	Compatible make	Toyota	522	0.161	21.47	31.75
	Compatible model	MR2 Spyder	8075	2.492	21.96	32.55
	Compatible year	2000	142	0.044	21.45	32.12
	Make	Toyota	51	0.016	83.77	97.13
	Type	Engine	1130	0.349	34.28	31.78
Computers	Brand	Dell	1591	0.123	0.57	0.84
	Color	Blue	424	0.033	51.04	48.59
	Compatible model	13 R2	2265	0.175	84.7	88.40
	Model	Inspiron	4897	0.379	52.07	45.32
	MPN	TXYDJ	7452	0.577	29.57	29.17
	Type	Motherboard	1567	0.121	15.7	24.48

Table 3: Comparison of approaches across features and metrics in the *sports* category.

Feature	Approach	Precision	Recall	Loose Precision	Loose Recall	Loose F1 Score	Coverage Improvement
Manufacturer	LLM	0.65	0.60	0.74	0.69	0.71	0.7/0.93
	Nearest Neighbors	**0.75**	**0.72**	**0.78**	**0.75**	**0.76**	**0.7/0.95**
	Extraction	0.52	0.47	0.60	0.55	0.57	0.7/0.93
Player	LLM	**0.80**	**0.79**	**0.86**	**0.86**	**0.86**	0.72/0.92
	Nearest Neighbors	0.68	0.69	0.74	0.74	0.74	**0.72/0.94**
	Extraction	0.57	0.60	0.66	0.70	0.68	0.72/0.93
Season	LLM	0.81	**0.83**	0.85	**0.88**	0.86	0.69/0.92
	Nearest Neighbors	0.63	0.61	0.70	0.68	0.69	**0.69/0.94**
	Extraction	**0.84**	**0.83**	**0.89**	0.87	**0.88**	0.69/0.91
Sport	LLM	0.79	0.75	0.89	0.84	0.86	**0.88/0.99**
	Nearest Neighbors	**0.91**	**0.91**	**0.95**	**0.95**	**0.95**	0.88/0.98
	Extraction	0.44	0.16	0.47	0.17	0.24	0.88/0.92
Team	LLM	0.54	0.41	**0.73**	0.55	0.63	0.64/0.82
	Nearest Neighbors	**0.59**	**0.68**	0.63	**0.73**	**0.67**	**0.64/0.95**
	Extraction	0.11	0.08	0.45	0.34	0.39	0.64/0.82
Type	LLM	0.08	0.08	0.43	0.43	0.43	**0.8/1.0**
	Nearest Neighbors	**0.75**	**0.71**	**0.90**	**0.85**	**0.87**	0.8/0.92
	Extraction	0.38	0.24	0.53	0.35	0.42	0.8/0.93

4.4 Results

Tables 3, 4, and 5 present the performance of three models families — LLM, Nearest Neighbors, and Data extraction across multiple metrics. The best results in each feature and metric are highlighted in bold. Coverage improvement measures how much the coverage of feature values was increased comparing to the values uploaded by the sellers (The number in the left side is the coverage of values uploaded by the sellers and the number in the right side is the coverage after completing missing values using our approaches).

General Trends. Across all categories, features with high distinct values and significant missing data (e.g., "Make" in *Motors*, "Compatible Model" and "MPN" in *Computers*) are harder for models to predict accurately, leading to lower performance in metrics like precision, recall, and coverage improvement. In contrast, features with fewer distinct values and moderate missing data benefit more from coverage improvements. LLMs generally provide better coverage for complex features like "MPN" and "Model" in *Computers* but often sacrifice precision. Nearest Neighbors maintains higher precision despite covering fewer data points, particularly for features like "Type" in *sports*.

Category-Specific Results. In *sports*, Nearest Neighbors outperforms other models in precision and recall for "Type" and "Sport," while LLM achieves the best results for "Player," a feature with many distinct values. Nearest Neighbors also shows high coverage improvement for "Sport" (0.88/0.99) and "Player" (0.72/0.94), while LLM struggles with features like "Team" and "Type." In the *Motors* category, high missing data (e.g., "Make" with 97.13% missing values) presents challenges. LLMs outperform in "Make" due to external knowledge, while Nearest Neighbors achieves better precision for "Compatible Make" and "Compatible Model," correlating with their moderate missing data and higher coverage improvement (e.g., 0.68/0.86 for "Compatible Make"). In the *Computers* category, LLMs excel in high-distinct-value features like "MPN" and "Model," achieving higher precision. Nearest Neighbors performs best for simpler features like "Brand" and "Color" but struggles with "MPN," where similar items may differ in this specific feature. The Extraction approach underperforms across all features.

4.4.1 *Error analysis.*

To better understand the limitations of our proposed methods for completing missing values, we conducted an error analysis across the categories. The results, visualized in Figure 3, reveal important insights into the patterns and sources of errors. Our analysis indicates that the vast majority of mistakes made by our methods can be attributed to cases where the correct values were entirely absent from the unlabeled data, which our methods are based on. When a value is not represented in this data, it becomes impossible for the model to predict it accurately. A smaller proportion of the errors occurred for values that were present in the unlabeled data but had fewer than ten occurences. This reflects the challenge of learning robust associations when data is sparse. Encouragingly, the

Table 4: Comparison of approaches across features and metrics in the *Motors* category.

Feature	Approach	Precision	Recall	Loose Precision	Loose Recall	Loose F1 Score	Coverage Improvement
Compatible make	LLM	0.44	0.44	0.56	0.57	0.56	0.68/0.83
	Nearest Neighbors	**0.46**	**0.53**	**0.68**	**0.79**	**0.73**	**0.68/0.86**
	Extraction	0.36	0.46	0.50	0.63	0.56	**0.68/0.86**
Compatible model	LLM	**0.22**	**0.21**	0.39	0.38	0.38	0.67/0.8
	Nearest Neighbors	0.17	**0.21**	**0.42**	**0.52**	**0.47**	0.67/0.86
	Extraction	0.09	0.18	0.28	**0.52**	0.36	**0.67/0.91**
Compatible year	LLM	**0.37**	**0.37**	0.41	0.41	0.41	0.68/0.78
	Nearest Neighbors	0.02	0.02	0.37	**0.49**	**0.42**	**0.68/0.86**
	Extraction	0.26	0.10	**0.50**	0.20	0.28	0.68/0.72
Make	LLM	**0.32**	**0.26**	**0.37**	**0.30**	**0.33**	**0.03/0.73**
	Nearest Neighbors	0.21	0.06	0.26	0.08	0.12	0.03/0.26
	Extraction	0.11	0.01	0.21	0.02	0.04	0.03/0.12
Type	LLM	0.18	0.19	**0.65**	**0.67**	**0.66**	**0.68/1.0**
	Nearest Neighbors	**0.23**	**0.22**	0.47	0.45	0.46	0.68/0.93
	Extraction	0.10	0.10	0.58	0.58	0.58	0.68/0.99

Table 5: Comparison of approaches across features and metrics in the *computers* category.

Feature	Approach	Precision	Recall	Loose Precision	Loose Recall	Loose F1 Score	Coverage Improvement
Brand	LLM	0.72	0.72	**0.79**	**0.80**	**0.79**	**0.99/1.0**
	Nearest Neighbors	**0.73**	**0.74**	0.77	0.79	0.78	**0.99/1.0**
	Extraction	0.48	0.49	0.55	0.57	0.56	**0.99/1.0**
Color	LLM	0.53	0.09	**0.69**	0.11	0.19	0.52/0.54
	Nearest Neighbors	**0.55**	**0.51**	0.60	**0.55**	**0.57**	**0.52/0.75**
	Extraction	0.04	0.03	0.22	0.17	0.20	**0.52/0.75**
Compatible model	LLM	**0.03**	**0.04**	**0.09**	**0.13**	**0.11**	**0.12/0.49**
	Nearest Neighbors	0.01	0.01	**0.09**	0.06	0.07	0.12/0.25
	Extraction	0.00	0.00	0.04	0.04	0.04	0.12/0.38
Model	LLM	**0.18**	**0.24**	**0.48**	**0.61**	**0.53**	**0.55/0.97**
	Nearest Neighbors	0.13	0.14	0.34	0.35	0.34	0.55/0.82
	Extraction	0.02	0.03	0.25	0.30	0.27	0.55/0.9
MPN	LLM	**0.37**	**0.22**	**0.45**	**0.27**	**0.34**	0.71/0.77
	Nearest Neighbors	0.09	0.16	0.12	0.20	0.15	**0.71/0.97**
	Extraction	0.05	0.10	0.09	0.19	0.12	0.71/0.85
Type	LLM	**0.35**	**0.36**	**0.68**	**0.70**	**0.69**	**0.76/1.0**
	Nearest Neighbors	0.26	0.23	0.53	0.47	0.50	0.76/0.86
	Extraction	0.17	0.16	0.48	0.45	0.46	0.76/0.98

methods performed well whenever sufficient data was available. For values that appeared frequently (e.g., more than 50 times), the models consistently completed the missing values accurately. This emphasizes the effectiveness of the proposed methods in leveraging information when it exists in a substantial amount. These observations highlight the dependency of data completion methods on the availability and distribution of the unlabeled data. They also underscore the importance of enhancing the training datasets, either by augmenting them with additional sources of information or by employing models capable of extrapolating knowledge to unseen values, such as LLM's.

4.4.2 Computational considerations. The computational costs of the three methods vary significantly, presenting a clear trade-off between efficiency and accuracy. The data extraction approach is the least computationally demanding, as it relies on statistical models to extract missing values from unstructured text, making it a fast and resource-efficient solution. However, its effectiveness depends on the quality and coverage of extracted terms, struggling with ambiguous or novel data. The nearest-neighbor approach using embedding similarity requires a moderate computational cost, as it leverages pre-trained models like CLIP to generate item embeddings and perform similarity searches. While this method provides improved accuracy over data extraction and does not require extensive retraining, it is computationally intensive due to embedding generation and nearest-neighbor computations. In contrast, LLMs offer the highest accuracy and contextual understanding but at a significantly higher computational cost. They require substantial resources for training and inference, making them less scalable for large e-commerce platforms. Ultimately, the choice of method depends on the trade-off between computational efficiency and the need for accurate feature completion.

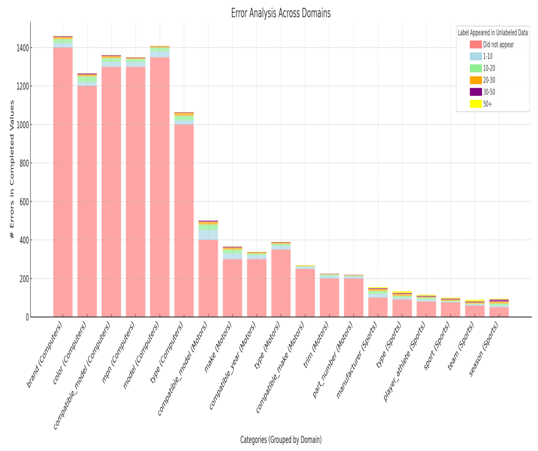


Figure 3: Error analysis for *sports*, *Motors* and *Computers*. Most of the errors in the completed values did not appear in the unlabeled data.

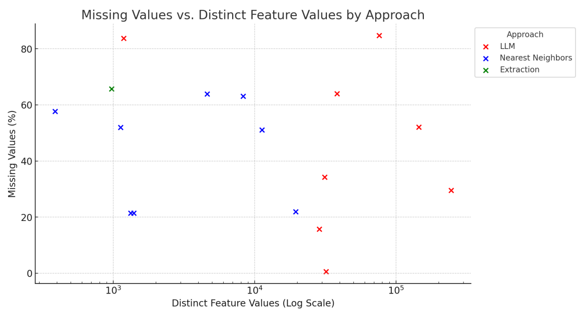


Figure 4: Missing Values vs. Distinct Feature Values by approach in the unlabeled dataset.

4.5 Case study

To demonstrate the value of completing missing values in e-commerce, we conducted a case study focusing on candidate retrieval and catalog matching in the *sports* category. Using a dataset of 5000 items, we retrieved similar items from the catalog for each listed item based on the title and feature values, both before and after completing missing values using our Nearest Neighbors approach. We compared the position of the correct catalog match within the retrieved list of candidates. While the position of the correct candidate remained unchanged for most items, in nearly 80% of the cases where a change occurred, the correct candidate was positioned higher in the list when the missing 'sport' feature—chosen for its high F-score in our experiments—was completed. This result highlights the impact of completing missing values on improving candidate retrieval accuracy, particularly for features with high predictive performance.

5 DISCUSSION

This paper makes several key contributions to the field of data completion in e-commerce. First, it presents and evaluates low-cost machine learning methods as effective alternatives to expensive large language models (LLMs) for completing missing values in product items. Second, it compares the performance of different approaches across various e-commerce categories, providing insights into when these low-cost methods are most effective. Lastly, the paper demonstrates that these low-cost methods can often achieve results comparable to LLMs, making them a practical solution for large-scale e-commerce platforms.

Figure 4 visually summarizes the results, showing the best-performing approach per feature, categorized by the number of distinct values and the percentage of missing data. The figure highlights that while LLMs are highly effective for handling features with a large number of distinct values, nearest neighbor approaches can achieve superior precision at a much lower computational cost for structured features with moderate distinct values. An exception is the red point in the left side of the graph, that represents a feature with very high precentage of missing values that causes the Nearest Neighbors approach to achieve lower performance than the LLM. This result underscores that low-cost methods can be a strong alternative to LLMs in many cases, particularly when dealing with repetitive, structured data. While LLMs excel in some scenarios, their legal, computational, and financial constraints often limit practical deployment. Fine-tuning or few-shot learning enhances domain knowledge but remains inferior to training on domain-specific data, which better captures platform-specific nuances. low-cost approaches are more scalable and cost-efficient in e-commerce.

Here are some guidelines when approaching a new category for choosing which group of methods to use: (1) Use Nearest Neighbors for features that are structured and have repeating values across items, such as manufacturer names, teams, or brands. This method effectively captures the similarity between items based on embeddings and yields strong performance in both precision and recall. (2) Opt for LLMs when dealing with features that are less structured and have diverse values, such as player names or specific product models. (3) When dealing with features that have high levels of missing data, LLMs appear to be a better approach than Nearest Neighbors, as they can leverage external knowledge not present in the dataset. (4) Fine-tune a model on the category's data to generate better embeddings, improving the accuracy of nearest neighbor methods for specific e-commerce categories.

6 SUMMARY AND FUTURE WORK

In this paper, we addressed the critical challenge of missing data in e-commerce items, focusing on low-cost machine learning approaches to complete missing feature values in comparison to large language model (LLM)-based methods. E-commerce platforms rely heavily on structured data. However, items from small sellers often lack complete data, making it harder to utilize these items effectively. To tackle this, we explored two primary approaches: *Data extraction* from unstructured text and the use of similar items through *Nearest Neighbor* methods based on embeddings.

We conducted extensive experiments using three diverse datasets — *sports trading cards, motors, and computers* - each with unique characteristics, ensuring that our methods were tested in different contexts. While this study shows promising results for low-cost machine learning methods in feature value completion, there are several avenues for further exploration: (1) Explore combining low-cost machine learning methods with LLMs to leverage the strengths of both. For example, LLMs could be used to enhance precision in specific features where traditional methods struggle, while lower-cost approaches can maintain efficiency. (2) Combining data sources to enrich the data (3) Scaling to Multimodal Data: Incorporating other modalities such as images could further improve missing data completion. (4) Improvements in Similarity Detection: Fine-tuning models to get embeddings for specific e-commerce categories may yield higher precision.

By addressing these areas, future research could not only improve the accuracy of feature completion methods but also enhance the practical applicability of these approaches in real-world e-commerce platforms.

REFERENCES

[1] Shahriar Akter and Samuel Fosso Wamba. 2016. Big data analytics in E-commerce: a systematic review and agenda for future research. *Electronic Markets* 26 (2016), 173–194.

[2] Sarah S Alrumiah and Mohammed Hadwan. 2021. Implementing big data analytics in e-commerce: Vendor and customer view. *Ieee Access* 9 (2021), 37281–37286.

[3] K Arya, T Kumar, and MK Jain. 2016. Big data analytics of global E-commerce organisations: A study survey and analysis. *Int. J. Sci. Eng. Res* 7, 12 (2016), 82–84.

[4] Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[5] Zhicheng Ding, Jiahao Tian, Zhenkai Wang, Jinman Zhao, and Siyang Li. 2024. Semantic Understanding and Data Imputation using Large Language Model to Accelerate Recommendation System. *arXiv preprint arXiv:2407.10078* (2024).

[6] Wafaa Mustafa Hameed and Nzar A Ali. 2023. Missing value imputation techniques: a survey. *UHD Journal of Science and Technology* 7, 1 (2023), 72–81.

[7] Ahatsham Hayat and Mohammad Rashedul Hasan. 2024. CLAIM Your Data: Enhancing Imputation Accuracy with Contextual Large Language Models. *arXiv preprint arXiv:2405.17712* (2024).

[8] Bernard J Jansen, Kholoud K Aldous, Joni Salminen, Hind Almerekhi, and Soon-gyo Jung. 2023. *Data Collection Methods. In: Understanding Audiences, Customers, and Users Via Analytics: An Introduction to the Employment of Web, Social, and Other Types of Digital People Data.* Springer.

[9] P. Liang et al. 2024. Understanding LLMs: A Comprehensive Overview from Training to Inference. *arXiv preprint arXiv:2401.02038* (2024). https://arxiv.org/abs/2401.02038

[10] Roderick JA Little and Donald B Rubin. 2019. *Statistical analysis with missing data.* Vol. 793. John Wiley & Sons.

[11] Ajinkya More. 2016. Attribute extraction from product titles in ecommerce. *arXiv preprint arXiv:1608.04670* (2016).

[12] Muhammad S Osman, Adnan M Abu-Mahfouz, and Philip R Page. 2018. A survey on data imputation techniques: Water distribution system as a use case. *IEEE Access* 6 (2018), 63279–63291.

[13] Duangmanee Putthividhya and Junling Hu. 2011. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing.* 1557–1567.

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning.* PMLR, 8748–8763.

[15] Daniel J Stekhoven and Peter Bühlmann. 2012. MissForest—non-parametric missing value imputation for mixed-type data. *Bioinformatics* 28, 1 (2012), 112–118.

[16] Yige Sun, Jing Li, Yifan Xu, Tingting Zhang, and Xiaofeng Wang. 2023. Deep learning versus conventional methods for missing data imputation: A review and comparative study. *Expert Systems with Applications* 227 (2023), 120201.

[17] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. 2001. Missing value estimation methods for DNA microarrays. *Bioinformatics* 17, 6 (2001), 520–525.

[18] Alexandru Tăbușcă. 2023. Tracing the Influence of Large Language Models across the Most Impactful Scientific Works. *Electronics* 12, 24 (2023), 4957. https://doi.org/10.3390/electronics12244957

[19] Stef Van Buuren and Karin Groothuis-Oudshoorn. 2011. mice: Multivariate imputation by chained equations in R. *Journal of statistical software* 45 (2011), 1–67.

[20] Zhenhua Wang, Olanrewaju Akande, Jason Poulos, and Fan Li. 2021. Are deep learning models superior for missing data imputation in large surveys? Evidence from an empirical comparison. *arXiv preprint arXiv:2103.09316* (2021).

[21] Jinsung Yoon, James Jordon, and Mihaela Schaar. 2018. Gain: Missing data imputation using generative adversarial nets. In *International conference on machine learning.* PMLR, 5689–5698.

[22] Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. Opentag: Open attribute value extraction from product profiles. In *Proc. of KDD.* 1049–1058.