Cost-Effective LLM Utilization for Machine Learning Tasks over Tabular Data

Yael Einy
Tel Aviv University
Israel
yaeleiny@mail.tau.ac.il

Tova Milo
Tel Aviv University
Israel
milo@cs.tau.ac.il

Slava Novgorodov
Tel Aviv University
Israel
slavanov@post.tau.ac.il

ABSTRACT

Classic machine learning (ML) models excel in modeling tabular datasets but lack broader world knowledge due to the absence of pre-training, an area where Large Language Models (LLMs) stand out. This paper presents an effective method that bridges the gap, leveraging LLMs to enrich tabular data to enhance the performance of classical ML models. Despite the previously limited success of direct LLM application to tabular tasks due to their high computational demands, our approach selectively enriches datasets with essential world knowledge, balancing performance improvement with cost-effectiveness. This work advances the capabilities of traditional ML models and opens new avenues for research at the convergence of classical ML and LLMs, marking the onset of a new era in cost-effective data enrichment.

CCS CONCEPTS

• **Information systems** → *Information integration.*

KEYWORDS

Data Enrichment, Data Integration, Large Language Models

ACM Reference Format:
Yael Einy, Tova Milo, and Slava Novgorodov. 2024. Cost-Effective LLM Utilization for Machine Learning Tasks over Tabular Data. In *Governance, Understanding and Integration of Data for Effective and Responsible AI (GUIDE-AI '24), June 09–15, 2024, Santiago, AA, Chile.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3665601.3669848

1 INTRODUCTION

In the realm of machine learning (ML), the analysis of tabular datasets is crucial for a wide range of applications, from financial forecasting to health diagnostics. Classic models like Random Forest and XGBoost have been pivotal in mastering classification and regression tasks on tabular data, delivering state-of-the-art performance, computationally efficient. However, these models face a significant challenge: their lack of pre-training restricts them from grasping the broader general knowledge that extends beyond the data.

This gap is precisely where pre-trained Large Language Models (LLMs) excel, indicating an untapped potential to enhance the modeling. This potential comes into sharper focus when considering that tabular datasets often lack essential information vital for accurate target variable modeling.

LLMs have revolutionized various domains. They excel at retrieving world knowledge and can also be fine-tuned with specific data to become experts in particular fields, like the data of a company. While their application to tabular data tasks has been explored [3, 7, 8], significant challenges persist. Direct application of LLMs to ML tasks on tabular data encounters inherent limitations due to the structured nature of tabular data. This often leads to LLMs being outperformed by simpler, more resource-efficient ML models.

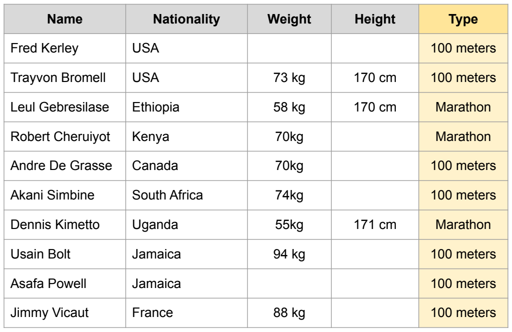
To address these gaps, we propose a novel method that efficiently leverages LLMs for enriching tabular datasets with relevant knowledge. The proposed method tackles the challenges of data incompleteness and contextual unawareness by economically utilizing LLMs to generate an extended set of features. Samples classified as 'hard' receive enrichment from a comprehensive array of base features, while 'easy' samples are processed with a narrower set. This novel selective enrichment process is designed to optimize LLM utilization, to improve the ML predictions. Let us illustrate the potential application of our method with a practical example.

Example 1.1. Consider a dataset of male athletes, both marathon and 100 meters runners, depicted in Figure 1. The dataset consists of information about athletes' names, nationalities, and occasionally their weight and height. We aim to predict whether an athlete is a sprinter or a marathon runner. The true value is depicted in the last column (marked in yellow) and not a part of the original data.[1]

GUIDE-AI '24, June 09–15, 2024, Santiago, AA, Chile

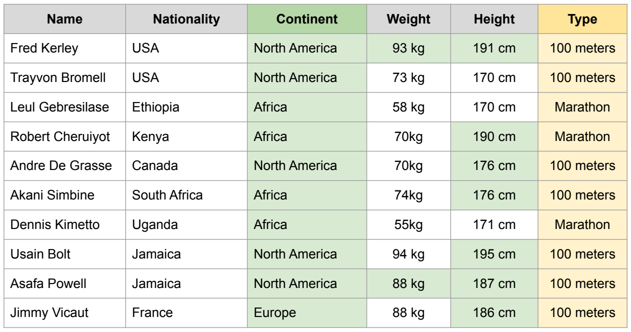
ACM ISBN 979-8-4007-0694-3/24/06
https://doi.org/10.1145/3665601.3669848

[1] This information is indeed missing even in the reliable sources such as Wikipedia. For example, as of April 2024, the page "Leul Gebresilase" on English Wikipedia has no information about weight and height and "Sara Hall" has no information about weight.



Name	Nationality	Weight	Height	Type
Fred Kerley	USA			100 meters
Trayvon Bromell	USA	73 kg	170 cm	100 meters
Leul Gebresilase	Ethiopia	58 kg	170 cm	Marathon
Robert Cheruiyot	Kenya	70kg		Marathon
Andre De Grasse	Canada	70kg		100 meters
Akani Simbine	South Africa	74kg		100 meters
Dennis Kimetto	Uganda	55kg	171 cm	Marathon
Usain Bolt	Jamaica	94 kg		100 meters
Asafa Powell	Jamaica			100 meters
Jimmy Vicaut	France	88 kg		100 meters

Figure 1: Athletes dataset - original



Name	Nationality	Continent	Weight	Height	Type
Fred Kerley	USA	North America	93 kg	191 cm	100 meters
Trayvon Bromell	USA	North America	73 kg	170 cm	100 meters
Leul Gebresilase	Ethiopia	Africa	58 kg	170 cm	Marathon
Robert Cheruiyot	Kenya	Africa	70kg	190 cm	Marathon
Andre De Grasse	Canada	North America	70kg	176 cm	100 meters
Akani Simbine	South Africa	Africa	74kg	176 cm	100 meters
Dennis Kimetto	Uganda	Africa	55kg	171 cm	Marathon
Usain Bolt	Jamaica	North America	94 kg	195 cm	100 meters
Asafa Powell	Jamaica	North America	88 kg	187 cm	100 meters
Jimmy Vicaut	France	Europe	88 kg	186 cm	100 meters

Figure 2: Athletes dataset - enhanced

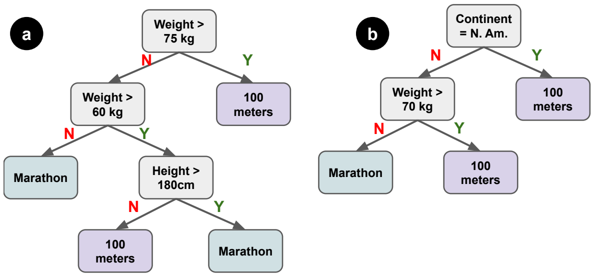


Figure 3: Potential models to classify the athletes.

Some attribute contents like weight and height that can be valuable for modeling are missing, whereas other information, such as age, which may be relevant, is not even part of the schema. Moreover, some existing attributes are sparse and inadequate for learning. To address these limitations, we use LLMs to infer and impute missing attribute values. Possible example of such imputation is depicted in Figure 2, with the extended information is marked in green.

A straightforward and somewhat non-realistic approach could be to fill all the missing cells and add a multitude of potentially relevant columns using the 'state-of-the-art' LLM. However, interactions with an LLM on a large scale, can become very costly, and hence one has to carefully consider which and how many questions to ask, as well as which LLM to use, as different LLMs can do similar enrichment at different costs. Thus, asking for all possible missing values or requesting each value of a completely new column, is unlikely to be cost-effective, especially using 'closed-book' LLMs (like GPT [12]), that are using their encoded knowledge, and cannot retrieve external data sources.

Typically, the expense of querying 'closed-book' LLMs correlates with the number of processed tokens, whereas other types of LLMs prioritize different cost objectives. For example, a type of LLM that can be cost effective for the task of introducing new columns is a Retrieval-Augmented Generation (RAG) model [10]. Unlike 'closed-book' LLMs, RAG models integrate retrieval mechanisms, enabling the fetching of external data sources to join the desired column. This means that instead of making multiple LLM queries for each data point, we might only need a single query (along with a retrieval operation) to add an entire column. The data sources can be retrieved from the Internet or from an internal corpus of a company. Another benefit of RAG models is their capability to access data sources released after their training, thus integrating current information into the dataset.

Transitioning from these capabilities, our system is designed to maximize gain based on improvements in ML model predictions, while staying within the cost budget. To select enrichment queries and the appropriate LLM for each, we assess the potential gain against the associated costs, which include querying the LLMs, processing tokens, and retrieving and processing external data, if required.

Moving on to the practical application of these strategies, Figure 3 depicts two simple models that can correctly classify the athletes based on the additional data. Model (a) heavily relies on the weight and height information and needs all these cells correctly filled while model (b) uses the continent information and does not need any additional information. Moreover, not all samples need to be augmented. For example, since there is not many Jamaican marathon runners, we could skip adding the continent information to the all the Jamaican athletes. The model then could be modified to start with the "Nationality = Jamaica" node, which would classify all the Jamaican athletes as 100 meters sprinters if the answer is yes and continue as model (b) if the answer is no.

To address the problem of cost-effective utilization of LLMs, we propose to build on previous work in the field of crowdsourcing (e.g., [2, 4]). Similarly to our setting, there the research problems often involved determining the optimal questions to pose to a crowd and identifying the most suitable individuals within the crowd to approach for specific tasks. This involves strategies to maximize the quality and relevance of responses. Other than the different practical application setting there is also a key difference in terms of the properties of the underlying optimization problem - the latency when interacting with LLMs, that respond almost instantly, is much lower. This allows for practical iterative solution, where each individual query can be decided after receiving the answer to the previous queries, whereas in crowdsourcing there is a human bottleneck that requires

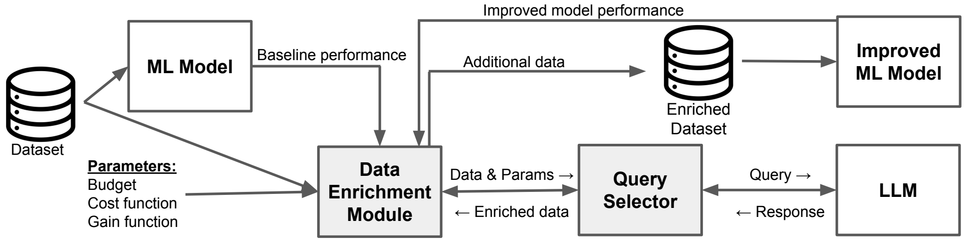


Figure 4: System architecture.

a more parallelized approach, that decided on the queries based on less information, thereby reducing the upper bound on optimization potential.

Contribution: We propose a novel, cost-effective framework for enriching tabular data using various LLMs to improve ML predictions, especially when the initial data is inadequate for effective learning.

2 TECHNICAL OVERVIEW

Next, we formally define the problem, followed by the system overview we propose and the research agenda.

2.1 Problem Definition

Dataset: Assume a tabular dataset D with n rows and d columns (features), $D = \{(x_i, y_i)\}_{i=1}^{n}$, where each x_i is a d-dimensional feature vector and y_i is a target value. We assume that each column has a natural-language name such as weight, height, or nationality.

LLM Query: We define several types of LLM queries[2] we use to enrich the data. First, we define the values-queries - queries that explicitly ask for a set of feature values for a given entity. An example of such a query is *"What is the nationality of Usain Bolt"*. We assume for simplicity that the response is retrieved in the following structured format: $\{``key_1" : ``value_1", ..., ``key_n" : ``value_n"\}$, hence the response to the query about Usain Bolt will be $\{``Nationality" : ``Jamaica"\}$.

A column query is a type of value query that generates values for a new feature across multiple entities in a single query. For instance, a column query might be *"Provide the GDP for each of the countries in* $\{``Italy", ``Chile", ...\}$*"*, will produce responses like $\{``Italy" : ``GDP1", ``Chile" : ``GDP2", ...\}$. Using RAG models for column queries is efficient because it leverages retrieval mechanisms to fetch and integrate the required data from external information.

Another type of query is a keys query, which is used to retrieve a possible set of features that we may want to add for a specific entity type. For example, such A query may be *"What are the three most useful features of countries, besides 'continent'?"*, with the response may be $\{``Feature1" : ``Population", ``Feature2" : ``Area", ``Feature3" : ``Currency", ...\}$.

Query cost: For all types of queries, we define the cost function of a single query, when pointed to a 'closed-book' LLM, M as:

$$cost(q, M) = c_q(M) + c_{tq}(M) \cdot \text{number\_of\_query\_tokens} \\ + c_{tr}(M) \cdot \text{number\_of\_response\_tokens}$$

In this definition, we "pay" $c_q(M)$ for the query itself and $c_{tq}(M)$, for each of the processed tokens in the query q, $c_{tr}(M)$ for each of the produced response tokens. The actual costs depends on the chosen LLM, M. This definition is inspired by the fact that the existing LLMs usually charge per tokens, more feature correlate with more tokens.

Using a RAG LLM involves retrieval cost $c_r(M)$, processing cost for $|D|$ documents $c_{dp}(M, |D|)$, and the prior factors:

$$cost(q, M) = c_q(M) + c_{tq}(M) \cdot \text{number\_of\_query\_tokens} \\ + c_r(M) + c_{dp}(M, |D|) \\ + c_{tr}(M) \cdot \text{number\_of\_response\_tokens}$$

Using RAG models, a single query can generate an entire column, reducing the number of queries needed. Thus, since processing response tokens with RAG models may be cheaper, using RAG for generating new columns likely results in a lower total cost despite retrieval expenses.

Gain: We define the gain of the additional set of feature values to dataset D as, for classification tasks as follows:

$$gain(D_{new}) = Accuracy(Model(D_{new})) - Accuracy(Model(D))$$

For regression tasks:

$$gain(D_{new}) = MSE(Model(D_{new})) - MSE(Model(D))$$

This simply shows the relative improvement of the model trained on the enriched data.

Task: Enriching the data with extra features that maximize the gain while maintaining the cost within a defined budget.

2.2 System Overview

The visionary system we propose comprises several interconnected modules, each serving a specific purpose to facilitate efficient data processing and improve the accuracy of the downstream ML tasks. Figure 4 depicts the modules of the

[2] We use the notion of query while describing the interaction with the LLM, while this may be sometimes referred as "propmt" or "request".

system. Next, we focus on the two main modules (marked in grey in the figure) and describe their operation in details.

Data Enrichment Module. This module is responsible for the interaction with ML models (both with the baseline and the improved models). It receives the data, the results from the ML model and the set of input parameters, such as cost and gain functions, together with the budget. The module interacts with the *Query Selector* and enriches the data that later passes to the improved ML model. This process is iterative until the budget limit is reached.

To address LLM hallucinations, the iterative Data Enrichment Module verifies at each phase that the retrieved feature genuinely enhances ML performance. Features which do not improve ML performance are excluded from the dataset.

Query Selector. This pivotal component is responsible for the interaction with the LLM to enrich data. This module receives the data and the parameters from the Data Enrichment module and decides which queries to send, and to which of the LLMs, based on various factors. For example, one of such factors may be the applicability of the feature across multiple ML tasks, thereby maximizing resource utilization. In this case the gain will be computed as the sum of gains over all the relevant ML tasks. The selection of the LLM is determined by the varying costs and levels of uncertainty associated with each.

2.3 Research Agenda

Next, we present several directions for implementation and further research of different modules of our visionary system.

Targeted values enrichment: Since not all samples or categories within a dataset contribute equally to the performance, and since the effectiveness of one enrichment step depends on the others, *Query Selector* needs to determine which of the samples and categories are most beneficial as a group. An example for similar approach is presented in [6]. The described system identifies groups of information gaps whose closure significantly uplifts outcome quality. Adapting this strategy, allows to dynamically recognize data segments whose mutual enhancement could boost the model accuracy.

Enrichment based on task difficulty: *Query Selector* segment the dataset into groups based on their difficulty. For example, such segmentation may be based on the countries that are "hard to classify" vs "easy to classify", which allows for applying a resource-conserving approach for enrichment.

Specifically, for countries that are considered easy to classify, we can afford a more economical approach. By aggregating samples on the country level, *Query Selector* can pose a single, broader query to the LLM for each country, extracting features or even likelihoods. This method reduces the cost by minimizing the number of required LLM queries. For example *Query Selector* can issue a query *"What is the likelihood that an athlete from Jamaica is a marathon runner?"* and add the returned value as a feature to the dataset.

For the "hard to classify" categories, *Query Selector* adopts a more granular strategy. Here, the queries are not only country-specific but may also incorporate additional dimensions, e.g., athlete's height. Such specificity aims to capture nuanced details, albeit at a higher query cost. This strategy is flexible, for instance, rather than querying for a single height, it might query for a range, balancing specificity with cost efficiency. For example *"What is the likelihood that an athlete from USA, whose height is 180 to 185 cm, is a 100 meters runner?"*.

Iterative vs batch querying approach: Since the cost of each query includes the c_q part, querying for more features in one query may reduce the overall cost. However, some of the retrieved features may not be needed, and moreover, the iterative approach allows to evaluate the relative gain of each retrieved feature and change the strategy based on the improved ML model performance. Hence, a hybrid approach that interleaves both iterative approach (were the features are retrieved one-by-one) and batch approach (were one query retrieves many features) is needed.

3 RELATED WORK

Transformer-based LLMs, such as BERT [5] and GPT [12], have revolutionized NLP. Recent efforts like the TabLLM study [8] have explored the application of fine-tuned LLMs to tabular data classification, through prompt-serialization techniques. However, the approaches fall short in terms of resource efficiency.

Previous works [9, 11] have utilized LLMs for tabular feature engineering, leveraging the original features to develop auto-ML frameworks. While these methods are explainable and computationally efficient, their resulting features are just a product of the original ones, as in traditional feature engineering. Consequently, their effectiveness is limited to datasets that are inherently rich in relevant information. Moreover, [7] explored data enrichment by using unstructured text from a knowledge-base (KB), to create tabular features, utilizing LLMs to create these features. However, the reliance on named entities, to match to entities in the KB, limits the applicability of this method. Concurrently, [1] and [3] have demonstrated the utility of LLMs in generating feature embeddings and producing realistic tabular data, respectively, showcasing the potential of LLMs to transcend the traditional contextual limitations of classic ML.

Finally, as previously mentioned, research on crowdsourcing (e.g., [2, 4]) examines settings similar to our, where one aims to determine the optimal questions to pose to a crowd and the most suitable individuals within the crowd to approach for specific tasks. However, there are several practical differences, most notably the latency, hence an adaptation is required for our setting.

REFERENCES

[1] Dimitris Bertsimas, Kimberly Villalobos Carballo, Yu Ma, Liangyuan Na, Léonard Boussioux, Cynthia Zeng, Luis R Soenksen, and Ignacio Fuentes. 2022. TabText: a Systematic Approach to Aggregate Knowledge Across Tabular Data Structures. *arXiv preprint arXiv:2206.10381* (2022).

[2] Rubi Boim, Ohad Greenshpan, Tova Milo, Slava Novgorodov, Neoklis Polyzotis, and Wang-Chiew Tan. 2012. Asking the Right Questions in Crowd Data Sourcing. In *ICDE*. https://doi.org/10.1109/ICDE.2012.122

[3] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. 2023. Language Models are Realistic Tabular Data Generators. In *ICLR*. https://openreview.net/pdf?id=cEygmQNOeI

[4] Joseph Chee Chang, Saleema Amershi, and Ece Kamar. 2017. Revolt: Collaborative crowdsourcing for labeling machine learning datasets. In *CHI*.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*. https://doi.org/10.18653/v1/N19-1423

[6] Ido Guy, Tova Milo, Slava Novgorodov, and Brit Youngmann. 2021. Improving Constrained Search Results By Data Melioration. In *ICDE*.

[7] Asaf Harari and Gilad Katz. 2022. Few-Shot Tabular Data Enrichment Using Fine-Tuned Transformer Architectures. In *ACL*.

[8] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. 2023. Tabllm: Few-shot classification of tabular data with large language models. In *AISTATS*.

[9] Noah Hollmann, Samuel Müller, and Frank Hutter. 2023. Large Language Models for Automated Data Science: Introducing CAAFE for Context-Aware Automated Feature Engineering. In *NeurIPS*.

[10] Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wentau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *NeurIPS*.

[11] Yin Lin, Bolin Ding, H. V. Jagadish, and Jingren Zhou. 2024. SMARTFEAT: Efficient Feature Construction through Feature-Level Foundation Model Interactions. In *CIDR*.

[12] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. In *OpenAI Blog*.