Pool-Search-Demonstrate: Improving Data-wrangling LLMs via better in-context examples

Changho Shin *
Department of Computer Science
University of Wisconsin–Madison
cshin23@wisc.edu

Joon Suk Huh *
Department of Computer Science
University of Wisconsin–Madison
jhuh23@wisc.edu

Elina Choi
Department of Statistics
University of Wisconsin–Madison
echoi39@wisc.edu

Abstract

Data-wrangling is a process that transforms raw data for further analysis and for use in downstream tasks. Recently, it has been shown that foundation models can be successfully used for data-wrangling tasks Narayan et al. [2022]. An important aspect of data-wrangling with LMs is to properly construct prompts for the given task. Within these prompts, a crucial component is the choice of in-context examples. In the previous study of Narayan et al. [2022], demonstration examples are chosen manually by the authors, which may not be scalable to new datasets. In this work, we propose a simple demonstration strategy that individualizes demonstration examples for each input by selecting them from a pool based on their distance in the embedding space. Additionally, we propose a post-processing method that exploits the embedding of labels under a closed-world assumption. Empirically, our embedding-based example retrieval and post-processing improve foundation models' performance by up to 84% over randomly selected examples and 49% over manually selected examples in the demonstration. Ablation tests reveal the effect of class embeddings, and various factors in demonstration such as quantity, quality, and diversity.

1 Introduction

Data wrangling is a line of processes to transform raw data to improve data quality for further data analysis and downstream tasks. While data wrangling requires data scientists and engineers to spot and handle the problems in the datasets, machine learning has successfully automated data wrangling tasks such as entity matching [Li et al., 2020b], data cleaning [Rekatsinas et al., 2017, Heidari et al., 2019], data transformation [He et al., 2018], and schema matching [Zhang et al., 2021], reducing human efforts for data wrangling.

Foundation models (FM)—large models pre-trained with massive datasets—turned out to be very versatile, showing solid performance in many downstream tasks. The pioneering work by Narayan et al. [2022] has shown that large language model can be used for data wrangling tasks. The gist of their method is casting data-wrangling tasks into NLP tasks with proper instructions and demonstrations.

*These authors contributed equally

Table Representation Learning Workshop at NeurIPS 2023.

While successful, many questions remain in this application. One such question is how to select in-context examples. Narayan et al. [2022] manually chooses in-context examples, which imposes additional tuning work on the user side and is not adaptive to the given example. In this work, we introduce a simple method to enhance in-context examples. The main idea of our method is to select demonstration examples from a demonstration pool, which is typically the training dataset or the validation dataset. While this may appear to require additional annotated datasets, in practice, demonstration examples can be constructed using already restored data in database. We hypothesize that the quantity, relevancy, and diversity of demonstration examples have a crucial impact on the quality of prompts. To verify our hypothesis, we designed a simple demonstration example retrieval method based on the embedding space—pool-search-demonstrate (PSD) method. As an additional contribution, we also suggest a class embedding approach as a post-processing method. Our experiments show that our method can improve data wrangling performance of large language models by as much as 84% over random example demonstrations and 49% over manually selected example demonstrations, respectively. In an ablation study, we observe

- Examples in demonstrations play a more important role in the larger language model.
- Given relevant examples with our methods, adding more demonstrations typically yields better results in the large model. However, the small model does not improve or get worse as the number of examples increases in the prompt.
- As the demonstration pool has more diverse examples, our method can choose better demonstration examples, resulting in better performance.

2 Related works

Data-wrangling with ML. As machine learning systems have been widely adopted, they have been actively used in data-wrangling tasks as well. Magellan [Konda et al., 2016] extracts feature vectors to obtain similarity and perform matching between different tables for entity matching. Ditto [Li et al., 2020b] finetunes BERT to perform entity matching tasks with entry serialization and data augmentation, which can be seen as an earlier version of data-wrangling with FMs. Holoclean [Rekatsinas et al., 2017] utilizes graphical models for data imputation and error detection. Holodetect [Heidari et al., 2019] casts error detection task as a few-shot classification task. IMP [Mei et al., 2021] finetunes language models for data imputation, by casting it as a classification task.

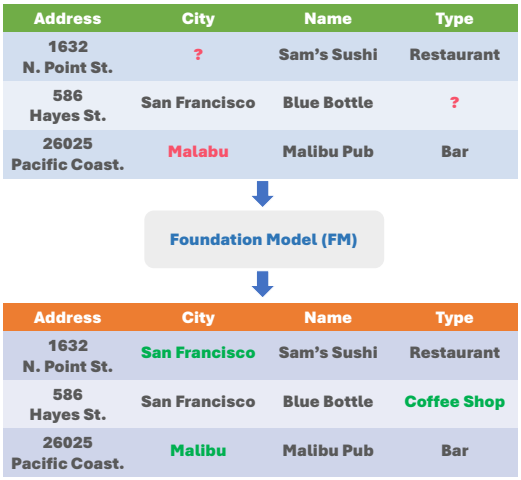

Figure 1: Illustration of data wrangling with foundation model. Foundation models can conduct various data wrangling tasks with proper instruction and demonstrations Narayan et al. [2022].

While such ML-based data-wrangling solutions have been actively studied, not many studies have been done regarding the application of FM to data-wrangling tasks. Narayan et al. [2022] is the first study of that, where foundation models are successfully applied to various data-wrangling tasks including entity matching, data imputation, error detection, and data transformation, beating previous SOTA algorithms. Vos et al. [2022] utilized prefix tuning Li and Liang [2021] to improve foundation models in data-wrangling tasks. Chen et al. [2023] used foundation models for data retrieval from multi-modal data lakes. Our study mainly builds up on the study of Narayan et al. [2022], focusing on the demonstration aspects of data-wrangling tasks.

Prompt engineering Prompt engineering or in-context prompting aims for effective learning of large language models (LLM) without updating the model weights. Zero-shot or few-shot learning, often considered to be the pioneer of different LLM methods, take naive approaches. Zero-shot learning consumes the task test to answer the desired results, such as sentiment or human intention. Few-shot learning, which is evaluated to work better than zero-shot, learns desirable "examples" consisting of both input and output. However, the performance of few-shot learning is sensitive

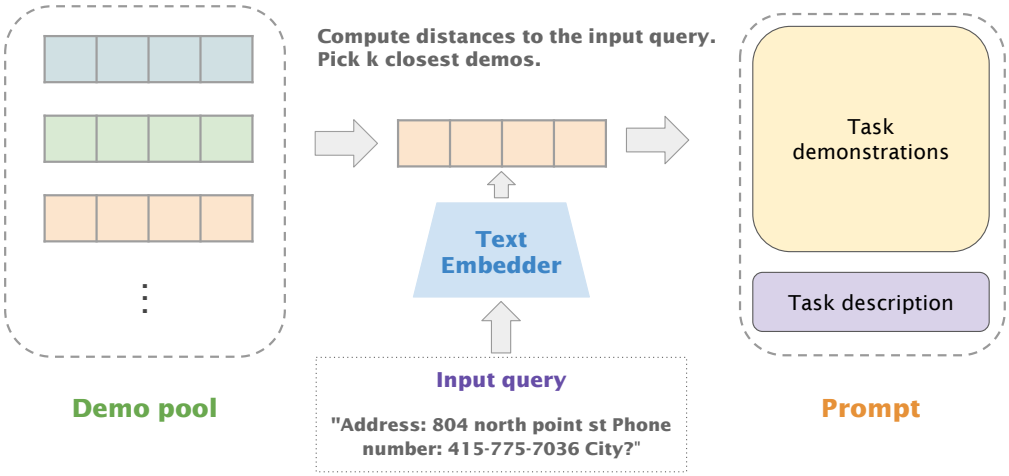

Figure 2: Proposed prompt construction method. First, we collect demonstration examples. Secondly, given the input data-wrangling task query, we get the embedding of the input query and find k closest demo examples from the demo pool. Finally, we construct an individualized prompt by putting together task demonstrations and a task description.

to the type of training examples and prompt format [Zhao et al., 2021]. Studies suggest different ways to improve example selection such as semantic approaches [Liu et al., 2021b], graph-based learning [Su et al., 2022], contrastive learning Rubin et al. [2021], Q-learning [Zhang et al., 2022], and uncertainty-based active learning [Diao et al., 2023]. Our method builds upon Liu et al. [2021b], which selects examples based on the similarity in the embedding space.

Demonstrations for in-context learning What makes good demonstrations is a practical issue in in-context learning, thus has been actively studied. Min et al. [2022] identified potential beneficial aspects of demonstrations, which are the input-label mapping, the distribution of the input text, the label space, and the format. They revealed that models can retain up to 95% of performance gains using either the inputs only or the label set only, if the proper format is used. Nonetheless, this insight can be applied only to small language sets, since the study of Wei et al. [2023] implies large language models actually can learn input-output pairs given in the prompt. Liu et al. [2021a] also studied the effect of variables in demonstrations, such as the number of examples in each prompt, the size of the pool for exemplar retrieval, and the order of in-context examples. We will include a similar analysis in the main experiments.

3 Improving FMs in data-wrangling tasks using embeddings

3.1 Solving data-wrangling tasks with FMs: data serialization

Narayan et al. [2022]'s study provides a basic setup to solve data-wrangling tasks with FMs; we keep the same framework. We mainly consider data imputation, entity matching, and error detection among many data-wrangling tasks. Data imputation is to infer the unknown value of an entry from other values in the entry. Entity matching is the task of deciding whether two different entries denote the same real-world entity. Error detection is the job of checking if there is an error in an entry.

Next, we formalize data-wrangling tasks. We denote the structured data set by D, a structured dataset with n entries, such that each entry is represented by a collection of m attribute-value pairs: for entry $e_i \in D$, $e_i = \{e_{i,j} = \{\mathsf{attr}_j, \mathsf{val}_j\} | 1 \leq j \leq m\}$. Based on these notations, each data-wrangling task can be expressed as:

- Data imputation (DI): given entry e and its unknown val_j corresponding to attr_j, DI model f_{DI} outputs a prediction $\widehat{\mathsf{val}}_j$, i.e.

$$f_{DI}(e, j) = \widehat{\mathsf{val}}_j.$$

- Entity matching (EM): given two entries e, e', EM model f_{EM} decides whether they denote the same real-world entity, i.e. $f_{EM}(e, e') \in \{\mathsf{True}, \mathsf{False}\}$.
- Error detection (ED): given entry e and a suspicious val_j, ED model f_{ED} detects if there is an error in val_j, i.e. $f_{ED}(e, j) \in \{\mathsf{True}, \mathsf{False}\}$.

We define a series of operations to cast data-wrangling tasks into NLP tasks. First, for the purpose of making an entry into a string, we define the serialize operation Narayan et al. [2022], Li et al. [2020a] as

$$\mathsf{serialize}(e) := \mathsf{attr}_1 : \mathsf{val}_1, \ldots, \mathsf{attr}_m : \mathsf{val}_m,$$

which is just a concatenation of attribute and value as a string. Next, task instruction strings T_l are defined for each task $l \in \{DI, EM, ED\}$ as follows

- $T_{DI}(e, j) := \mathsf{attr}_1 : \mathsf{val}_1, \ldots, \mathsf{attr}_j :?$
- $T_{EM}(e, e') :=$ Product A is $\mathsf{serialize}(e)$. Product B is $\mathsf{serialize}(e')$. Are Product A and Product B the same?
- $T_{ED}(e, j) :=$ Is there an error in val_j?

Accordingly, let $\tilde{e}, \tilde{e}'$ be examples, then demonstration strings are defined as

- $\mathsf{Demo}_{DI}(\tilde{e}, j) := T_{DI}(\tilde{e}, j), \mathsf{val}_j^*$, where val_j^* is a value of attr_j.
- $\mathsf{Demo}_{EM}(\tilde{e}, \tilde{e}') := T_{EM}(\tilde{e}, \tilde{e}'), y^*$.
- $\mathsf{Demo}_{ED}(\tilde{e}, j) := T_{ED}(\tilde{e}, j)\mathsf{val}_j?, y^*$.

Here val_j^* and $y^* \in \{\mathsf{True}, \mathsf{False}\}$ are ground truth.

Let $\mathsf{LM}(\cdot)$ be the language model output function given by an FM. Finally, an FM's data-wrangling task functions are given by

- Data imputation: $f_{DI}^r(e) := \mathsf{LM}\big(\mathsf{Demo}_{DI}(\tilde{e}_1, j), \ldots, \mathsf{Demo}_{DI}(\tilde{e}_r, j), T_{DI}(e, j)\big)$.
- Entity matching: $f_{EM}^r(e, e') := \mathsf{LM}\big(\mathsf{Demo}_{EM}(\tilde{e}_1, \tilde{e}_1'), \cdots, \mathsf{Demo}_{EM}(\tilde{e}_r, \tilde{e}_r'), T_{EM}(e, e')\big)$.
- Error detection: $f_{ED}^r(e, j) := \mathsf{LM}\big(\mathsf{Demo}_{ED}(\tilde{e}_1, j), \ldots, \mathsf{Demo}_{ED}(\tilde{e}_r, j), T_{ED}(e, j)\big)$.

Here r is the number of demonstrations.

Based on the formalism above, we propose our method in the following subsection.

3.2 Pool-Search-Demonstrate

In comparison to Narayan et al. [2022], we construct an *example pool* for demonstration, which will guide FMs to perform well on given tasks. The main consideration for the example pool is diversity. For the sake of simplicity, we first explain how to construct an example pool for data imputation (DI) and error detection (ED) tasks.

We start from k centroids of the dataset. Given n entries $\mathcal{E} = \{e^1, \ldots, e^n\}$, denote embedded entries as $x^i = \mathsf{Embedder}(\mathsf{serialize}(e^i))$ for $i \in [n]$, and embedded dataset as $\mathcal{X} = \{x^1, ..., x^n\}$. We get centroids by k-means, i.e.

$$(\mathsf{centroidX},\ \mathsf{centroidE}) = \mathsf{kMeans}(\mathcal{X}, \mathcal{E}, k),$$

where centroidX is the centroids in the embedding space and centroidE is the corresponding entries. The example pool is constructed as

$$\mathsf{Pool}(\mathcal{E}) := \big\{(e, y_e)\big|\ e \in \mathsf{centroidE},\ y_e \text{ is a label for } e\big\},$$

where the task-dependent labels $\{y_e\}$ are either provided from a human annotator or given in a dataset. In practice, if the annotated dataset is small, the example pool can be constructed by the whole dataset.

Given target entry e, we define the demonstration set M_e of the size m such that

$$M_e := \mathsf{kNN}\big(x, \mathsf{Pool}(\mathcal{E}), m\big),\ \ x := \mathsf{Embedder}\big(\mathsf{serialize}(e)\big),$$

that is, we select m tuples from $\mathsf{Pool}(\mathcal{D})$ that are close to the target e in the embedding space. Intuitively speaking, parameter k is related to diversity, and m controls quantity and relevancy.

Finally, for a given task $l \in \{DI, ED, \dots\}$, the prompt for e is

$$\mathsf{Prompt}_l(e) := \mathsf{Demo}_l(M_e), T_l(e),$$

where Demo is a method to generate a prompt based on M_e. For example, in the data imputation task,

$$\mathsf{Demo}_{DI}(M_e) = \mathsf{attr}_1^1 : \mathsf{val}_1^1 \cdots \mathsf{attr}_j^1 :? \; y^1, \dots, \\ \mathsf{attr}_1^m : \mathsf{val}_1^m \cdots \mathsf{attr}_j^m :? \; y^m.$$

Here attr_j^i represents the attribute j of $i-$th entry in M_e, and val_j^i represents the corresponding value.

For entity matching (EM) tasks, examples and inputs are pairs of data entries. The above procedure can be generalized to EM tasks by using $x = \mathsf{Embedder}(\mathsf{serialize}(e, e'))$ as the embedding for a pair (e, e'). By applying kMeans and kNN to those embeddings, one can construct a demonstration set $M_{(e,e')}$ for input (e, e') to an EM task. The prompt is generated by prefixing $T_{EM}(e, e')$ with a demonstration prompt generated from $M_{(e,e')}$.

Expected results Based on our hypotheses, the expected results are as follows.

- **Quantity**: As the number of demonstration examples m increases, the performance improves.
- **Relevancy**: kNN-based retrieval from the example pool will be better than random sampling.
- **Diversity**: As the number of centroids k increases, the performance improves. Also, if m is the same, selection from centroids will be better than random sampling or fixed examples.

3.3 Class embedding

While the output space of FM can be written as $\bigcup_{k=1}^{M} \mathcal{C}^{(k)}$, where M is the maximum number of tokens in output, $\mathcal{C}$ is the set of tokens and $\mathcal{C}^{(k)}$ denotes the k-fold Cartesian product of $\mathcal{C}$, typical data-wrangling tasks have smaller label space than that. With the closed world assumption Konda et al. [2016], we restrict the output space of FM to a finite label space to achieve better odds of correctness. Assuming the finite label space $\mathcal{Y}$ is known (for example $\mathcal{Y} = \{\text{Yes}, \text{No}\}$ for error detection or entity matching tasks), we construct a labeling map $L_Y : \bigcup_{k=1}^{M} \mathcal{C}^{(k)} \to \mathcal{Y}$ using an embedding map $\mathsf{Embedder} : \bigcup_{k=1}^{M} \mathcal{C}^{(k)} \to \mathbb{R}^d$ where d is the embedding dimension, as follows: Given an FM output $o = \mathsf{LM}(\mathsf{Prompt}_l(e)) \in \bigcup_{k=1}^{M} \mathcal{C}^{(k)}$, the labeling map $L_Y : \bigcup_{k=1}^{M} \mathcal{C}^{(k)} \to \mathcal{Y}$ is such that

$$L_Y(o) \in \arg\min_{y \in \mathcal{Y}} \left\| \mathsf{Embedder}(o) - \mathsf{Embedder}(y) \right\|_2,$$

where $\| \cdot \|_2$ is the Euclidean norm in $\mathbb{R}^d$.

We call this proposed labeling strategy as the class embedding labeling or simply "class embedding". The embedding map can in general be an another transformer trained to give semantic embeddings of sentences. Intuitively, if the embedding map captures the underlying semantics correctly, it would give rise to a superior labeling performance than a hard-coded rule-based labeling map due to it's resilence to perturbations. For example: If a task's correct label is $y =$ "$Samsung$" while FM's output is $o =$ " : $Samseong, ,$", it is likely that hard-coded rules will fail but a semantic embedder will place $o =$ " : $Samseong, ,$" closer to $y =$ "$Samsung$" than $y =$ "$Apple$".

Expected results We expect the class embedding strategy can enhance the performance proportionally to the size of $\mathcal{Y}$ since FMs can easily learn small label spaces while its output can be noisier when $\mathcal{Y}$ is large. Practically, it will give much improvement on data imputation tasks, where labels can have various values based on its task, while it will give a marginal improvement in error detection or entity matching tasks since there are only 2 possible labels and they can be inferred with demonstrations.

4 Experiment results

4.1 Experiment setup

Task & dataset. We benchmarked our method over the following data-wrangling tasks: data imputation, error detection, and entity matching. For data imputation, we use Buy and Restaurants

		Data imputation (Acc.)		Entity matching (F1)		Error detection (F1)	
Model	Demo. method	Buy	Restaurant	DBLP-ACM	Walmart	Hospital	Adult
Prev.ML Best	X	96.5	77.2	99.0	86.8	94.4	99.1
FLAN-T5-3B	random	98.5	65.1	81.3	**78.2**	10.9	**58.5**
	manual	98.5	**88.4**	78.0	75.9	15.5	0
	PSD	**100.0**	77.9	**89.3**	72.7	**67.4**	34.6
Vicuna-7B	random	98.5	80.2	86.5	68.7	4.6	58.0
	manual	**100.0**	**87.2**	73.1	59.0	30.7	38.0
	PSD	**100.0**	81.4	**97.5**	**82.5**	**89.4**	**87.3**

Table 1: Best performance varying the number of demonstrations, clusters, and class embedding. Prev. ML Best model represented traditional ML models for data-wrangling tasks, which appeared in Narayan et al. [2022]. According to them, IMP [Mei et al., 2021] in data imputation, Ditto [Li et al., 2020b] in entity matching, HoloDetect[Heidari et al., 2019] in error detection, worked best respectively among traditional ML approaches. The best scores in each model are bold.

[Mei et al., 2021]. DBLP-ACM and Walmart datasets are used for entity matching [Konda et al., 2016]. Finally, Hospital and Adult datasets are used in the error detection task. For figures-of-merits, the accuracy is used in data imputation, and the F1 score is used in entity matching and error detection.

Model. For foundational models, we used FLAN-T5 3B [Chung et al., 2022] and Vicuna 7B [Chiang et al., 2023]. FLAN-T5 is a Google's open-source language model, which is an encoder-decoder model based on the transformer Vaswani et al. [2017]. It is the same architecture with Raffel et al. [2020], but the main difference is that it employs the instruction-based fine-tuning, which is a fine-tuning based on various downstream tasks after pre-training. Note that the instruction-based fine-tuning is the main technique behind GPT3 and GPT3.5 Brown et al. [2020]. Vicuna is an open-source FM trained by fine-tuning LLaMA Touvron et al. [2023] on user-shared conversations collected from ShareGPT. Our main baseline is the "manual" demonstration which was crafted in Narayan et al. [2022]. Detailed experiment setups are given in Appendix A.1.

4.2 Main result

We report the best performance varying the number of demonstrations, clusters, and class embeddings given each model and demonstration methods in Table 1. Here, as a performance reference, we provide existing ML methods for data-wrangling tasks Narayan et al. [2022]. The best-performing parameters can be found in Appendix A.2 and the demonstration examples can be found in Appendix A.4.

We summarize general insights from the best-result settings. First of all, we observe that our method often yields performance comparable to or better than the best of previous ML or FM-based algorithms, even with much smaller models than GPT3-175B used by Narayan et al. [2022]. At the same time, our method's performance scales well with respect to the model size, especially, it generally performs better with Vicuna-7B than FLAN-T5-3B. Secondly, as expected, the class embedding method boosts performances in most cases. Thirdly, while typically more demos yield better results, they occasionally worsen the performance. However, we observe that having at least one demo was important. Lastly, contrary to our expectation, PSD worked better with a full demo pool of training data than the pool of centers in k-means clustering in most cases. Second - fourth points will be discussed in the following ablation study.

4.3 Ablation study

4.3.1 Class embedding

While class embedding turned out to be effective in various settings by replacing the post-processing of FM outputs, the effectiveness of this method varies depending on the domain and dataset. Figure 4 provides the average gain obtained by class embedding in experiment settings in our configuration grid. Interestingly, we can observe that the average gain increases proportionally to the cardinality of the label space. Based on this, we can suggest future work studying the relationship between the

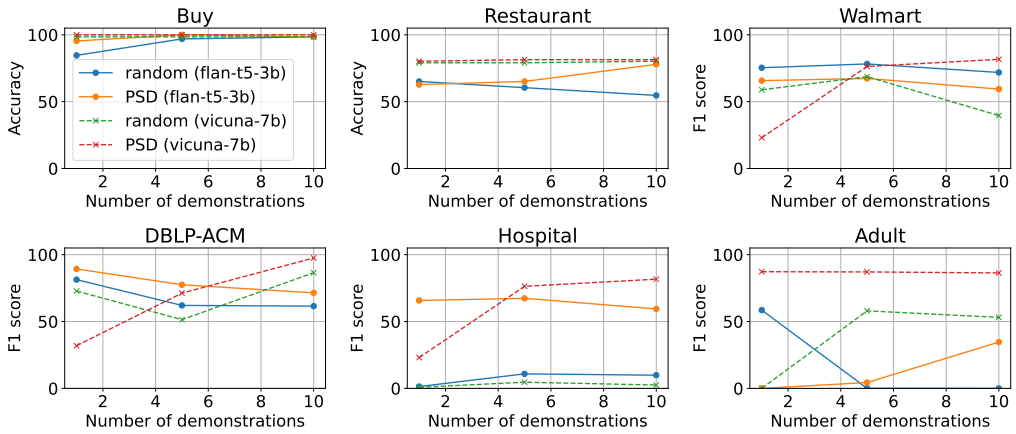

Figure 3: With a full demonstration pool, we report performance results by varying models, sampling methods (random and PSD), and the number of demonstrations. We can observe ablation study results about model, quantity, and relevancy.

cardinality of the label space and the effectiveness of class embedding in more general settings, i.e. text classification, text generation, etc.

4.3.2 Model size & Quantity

Though we only used two classes of models — FLAN-T5-3B and Vicuna-7B, we were able to observe a tendency that the larger model (Vicuna-7B) gets better with more demonstrations in Figure 3. The performance of FLAN-T5-3B tends to be relatively flat related to the number of demonstrations, while the performance scores of Vicuna-7B improved as the number of demonstrations increased given proper demonstrations (PSD and manual). This observation is concurrent with the existing works related to in-context learning Garg et al. [2022], Wies et al. [2023], Wei et al. [2023], where larger models actually can learn better from the in-context examples, while smaller models mainly use their prior learned from pre-training rather than in-context examples.

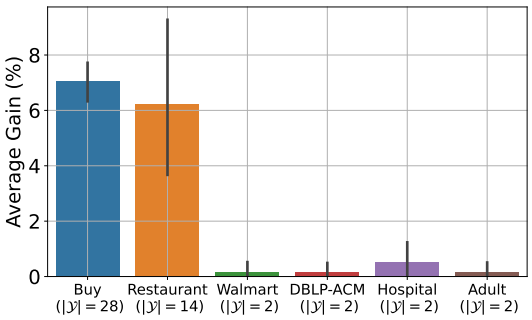

Figure 4: Average gain by class embedding in scores, depending on the dataset, $|\mathcal{Y}|$ denotes the cardinality of label space in each dataset.

4.3.3 Relevancy

By comparing the random demo and PSD in Figure 3, we observe how relevancy affects performance. In most cases, PSD gives better results from the beginning (Num. demo=1) or PSD's performance gradually increases to crossover that of the random demo's. This tendency is more clearly visible for larger Vicuna-7B model, which implies examples' relevancy is a key performance factor in larger language models.

4.3.4 Diversity

We hypothesized that k-means clustering in the demonstration example pool can give a diversity of demonstrations, which eventually improves the final performance scores. We verify the idea in Figure 5, where model is fixed as Vicuna-7B and the demonstration method is fixed as PSD, and the number of examples and the number of clusters are varied. As we expected, the performance scores improve as the number of clusters increases in most cases. Especially, in most cases, k-means clustering is

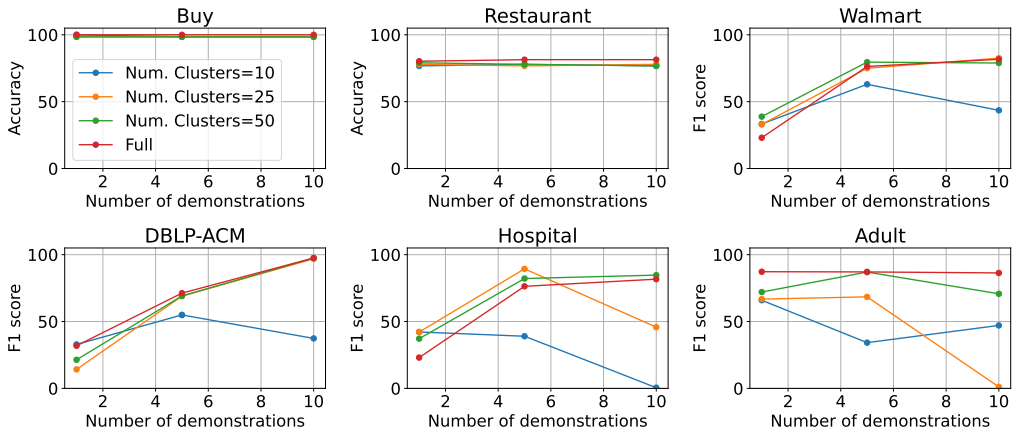

Figure 5: With the fixed model (Vicuna-7B) and demonstration method (PSD), we show performance results by varying number of demonstrations and number of clusters. We can observe ablation study results about diversity.

unnecessary if we only consider model's performance — using all of the training dataset examples yields the best results in most cases. Still, using 50 centroids can give comparable results to a full demonstration pool. Thus, in the case that keeping all of examples and trying retrieval can be costly, practitioners may consider clustering to reduce the cost.

5 Conclusion

In this project, we studied a demonstration strategy to improve FMs' performances in data-wrangling tasks. Specifically, our embedding-based example selection and post-processing could improve FM performance up to 84% over random examples and 49% over manually selected examples in the demonstration. Our contributions can be summarized as follows.

- We re-confirmed the idea that FMs can be used for data-wrangling tasks, consistently with Narayan et al. [2022], even with smaller models.
- We proposed an embedding-based example selection method, significantly improving FMs' performances across various datasets.
- We proposed a post-processing method based on embedding as well, which especially gives a significant advantage for the tasks where the cardinality of label space is finite and relatively large.
- We re-confirmed previous observations regarding demonstration, especially in data-wrangling tasks, by ablation test. As previous studies imply, the demonstration is more important for larger modelsWei et al. [2023], and the relevancy of examples improves the effectiveness of prompts Liu et al. [2021a].

Acknowledgement

We thank Prof. Paris Koutris and Prof. Frederic Sala (UW–Madison) for their comments on an earlier version of the manuscript, that greatly improved the manuscript. We would also like to show our gratitude to 3 "anonymous" reviewers for their insightful comments.

References

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Z. Chen, Z. Gu, L. Cao, J. Fan, S. Madden, and N. Tang. Symphony: Towards natural language query answering over multi-modal data lakes. In *Conference on Innovative Data Systems Research, CIDR*, pages 8–151, 2023.

W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://vicuna.lmsys.org`.

H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

S. Diao, P. Wang, Y. Lin, and T. Zhang. Active prompting with chain-of-thought for large language models. *arXiv preprint arXiv:2302.12246*, 2023.

S. Garg, D. Tsipras, P. S. Liang, and G. Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

Y. He, X. Chu, K. Ganjam, Y. Zheng, V. Narasayya, and S. Chaudhuri. Transform-data-by-example (tde) an extensible search engine for data transformations. *Proceedings of the VLDB Endowment*, 11(10):1165–1177, 2018.

A. Heidari, J. McGrath, I. F. Ilyas, and T. Rekatsinas. Holodetect: Few-shot learning for error detection. In *Proceedings of the 2019 International Conference on Management of Data*, pages 829–846, 2019.

P. Konda, S. Das, A. Doan, A. Ardalan, J. R. Ballard, H. Li, F. Panahi, H. Zhang, J. Naughton, S. Prasad, et al. Magellan: toward building entity matching management systems over data science stacks. *Proceedings of the VLDB Endowment*, 9(13):1581–1584, 2016.

X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Y. Li, J. Li, Y. Suhara, A. Doan, and W.-C. Tan. Deep entity matching with pre-trained language models. *Proceedings of the VLDB Endowment*, 14(1):50–60, 2020a.

Y. Li, J. Li, Y. Suhara, A. Doan, and W.-C. Tan. Deep entity matching with pre-trained language models. *arXiv preprint arXiv:2004.00584*, 2020b.

J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*, 2021a.

X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, and J. Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021b.

Y. Mei, S. Song, C. Fang, H. Yang, J. Fang, and J. Long. Capturing semantics for imputation with pre-trained language models. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 61–72. IEEE, 2021.

S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*, 2022.

A. Narayan, I. Chami, L. Orr, and C. Ré. Can foundation models wrangle your data? *arXiv preprint arXiv:2205.09911*, 2022.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

T. Rekatsinas, X. Chu, I. F. Ilyas, and C. Ré. Holoclean: Holistic data repairs with probabilistic inference. *arXiv preprint arXiv:1702.00820*, 2017.

O. Rubin, J. Herzig, and J. Berant. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*, 2021.

H. Su, J. Kasai, C. H. Wu, W. Shi, T. Wang, J. Xin, R. Zhang, M. Ostendorf, L. Zettlemoyer, N. A. Smith, et al. Selective annotation makes language models better few-shot learners. *arXiv preprint arXiv:2209.01975*, 2022.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

D. Vos, T. Döhmen, and S. Schelter. Towards parameter-efficient automation of data wrangling tasks with prefix-tuning. In *NeurIPS 2022 First Table Representation Workshop*, 2022.

J. Wei, J. Wei, Y. Tay, D. Tran, A. Webson, Y. Lu, X. Chen, H. Liu, D. Huang, D. Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

N. Wies, Y. Levine, and A. Shashua. The learnability of in-context learning. *arXiv preprint arXiv:2303.07895*, 2023.

J. Zhang, B. Shin, J. D. Choi, and J. C. Ho. Smat: An attention-based deep learning solution to the automation of schema matching. In *Advances in Databases and Information Systems: 25th European Conference, ADBIS 2021, Tartu, Estonia, August 24–26, 2021, Proceedings 25*, pages 260–274. Springer, 2021.

Y. Zhang, S. Feng, and C. Tan. Active example selection for in-context learning. *arXiv preprint arXiv:2211.04486*, 2022.

Z. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR, 2021.

Tasks	Metric	Dataset	N_{train}	N_{test}
Data imputation	Accuracy	Restuarant	469	65
		Buy	622	86
Entity matching	F1 score	Walmart-Amazon	6144	2049
		DBLP-ACM	7414	2473
Error detection	F1 score	Hospital	1710	17101
		Adult	11000	11000

Table 2: Summary of real datasets

A Experiment details

A.1 Detailed experiment setup

Datasets Details of datasets are described in Table 2. We evaluate models on test datasets and demonstration pools are constructed from the training set.

Prompts Given data entry e (attribute-data pairs), we used the following instruction sets for each task.

- Data imputation: $T_{DI}(e, j) = \mathsf{attr}_1 : \mathsf{val}_1 ... \mathsf{attr}_j :?$
  E.g. *name: girafe. addr: 208 e. 58th st. between 2nd and 3rd aves.. phone: 212/752-3054. type: italian. What is the city?* (Restaurant dataset)
- Entity matching: $T_{EM}(e, e') =$A is serialize(e). B is serialize(e'). Are A and B the same?
  E.g. *Product A is title: heavy duty 3-hole punch. modelno: 997. Product B is title: heavy-duty punch. modelno: nan. Are A and B the Same?* (Walmart dataset)
- Error detection: $T_{ED}(e, j) =$Is there an error in $\mathsf{attr}_j : \mathsf{val}_j$?
  E.g. *Is there a x spelling error in CountyName: etowxh?* (Hospital dataset)

Each demonstration has the answer after the above instruction formats.

Parameter search space For the purpose of parameter tuning and ablation test, we considered the following parameter space in summary.

- Model: [‘flan-t5-3b’, ‘vicuna-7b’]
- Demonstration method: [‘random’, ‘manual’, ‘PSD’] // PSD: Pool-Search-Demonstration
- Num. demos: [0, 1, 5, 10] // manual only has 10 demos
- Num. clusters (PSD only): [10, 25, 50, full (use all train data as the search space)]
- Class embedding: [True, False]

A.2 Details of Table 1

Table 3 show parameters used in Table 1 results.

A.3 Comparison with GPT3-175B

Table 4 shows the comparison of our methods with GPT3-175B. PSD yields significant improvements for small models so that they have performance comparable to GPT3-175B.

A.4 Demonstration comparison

Here we provide examples of demonstrations.

1. **Buy**

		Data imputation (Acc. ↑)			
	Demo. method	Buy		Restaurant	
		Num. Demos	Num. Clusters	Num. Demos	Num. Clusters
flan-t5-3b	random	10	-	1	-
	PSD	5	full	10	full
vicuna-7b	random	10	-	10	-
	PSD	10	full	10	full
		Entity matching (F1 ↑)			
	Demo. method	DBLP-ACM		Walmart	
		Num. Demos	Num. Clusters	Num. Demos	Num. Clusters
flan-t5-3b	random	1	-	5	-
	PSD	1	full	10	10
vicuna-7b	random	10	-	5	-
	PSD	10	full	10	25
		Error detection (F1 ↑)			
	Demo. method	Hospital		Adult	
		Num. Demos	Num. Clusters	Num. Demos	Num. Clusters
flan-t5-3b	random	5	-	1	-
	PSD	5	full	10	full
vicuna-7b	random	5	-	5	-
	PSD	5	25	1	full

Table 3: Best parameter settings used in Table 1

		Data imputation (Acc. ↑)		Entity matching (F1 ↑)		Error detection (F1 ↑)	
Model	Demo. method	Buy	Restaurant	DBLP-ACM	Walmart	Hospital	Adult
Prev.ML Best	X	96.5	77.2	99.0	86.8	94.4	99.1
GPT3-175B	X	84.6	70.9	93.5	60.6	6.9	0
	manual	98.5	88.4	96.6	87.0	97.8	99.1
flan-t5-3b	random	98.5	65.1	81.3	**78.2**	10.9	**58.5**
	manual	98.5	**88.4**	78.0	75.9	15.5	0
	PSD	**100.0**	77.9	**89.3**	72.7	**67.4**	34.6
vicuna-7b	random	98.5	80.2	86.5	68.7	4.6	58.0
	manual	**100.0**	**87.2**	73.1	59.0	30.7	38.0
	PSD	**100.0**	81.4	**97.5**	**82.5**	**89.4**	**87.3**

Table 4: Comparison with GPT3-175B

- **Input**: "name: Onkyo TX-NR906 A/V Receiver - TXNR906B. description: THX Ultra 2, Dolby Digital Plus, Dolby TrueHD, DTS-HD, Neural SurroundFM, AM. Who is the manufacturer?" (Answer: Onkyo)
- **Random demo**: "name: Sony alpha DSLR-A350K Digital SLR Camera with 18-70mm Zoom Lens - Black - DSLRA350K. description: 14.2 Megapixel - 3.9x Optical Zoom - 2.7' Active Matrix TFT Color LCD. Who is the manufacturer? Sony"
- **PSD (Num. clusters: 50)**: "name: Onkyo TX-SR876 A/V Receiver - TXSR876B. description: Dolby Digital, Dolby TrueHD, DTS-HD, Dolby Digital EX, Dolby Digital Plus, Dolby Pro Logic II, Dolby TrueHD, DTS, DTS-ES, DTS Neo:6, THX Surround EX, THX Select 2FM, AM. Who is the manufacturer? Onkyo"

2. **Restaurant**
   - **Input**: "name: chez michel. addr: 804 northpoint. phone: 415/775-7036. type: french. What is the city?" (Answer: san francisco)
   - **Random demo**: "name: anthonys. addr: 3109 piedmont rd. just south of peachtree rd.. phone: 404/262-7379. type: american. What is the city? atlanta"
   - **PSD (Num. clusters: 50)**: "name: fleur de lys. addr: 777 sutter st.. phone: 415-673-7779. type: french (new). What is the city? san francisco"
3. **DBLP-ACM**
   - **Input**: "Product A is title: secure transaction processing in firm real-time database systems. authors: binto george , jayant r. haritsa. venue: sigmod conference. year: 1997. Product B is title: secure buffering in firm real-time database systems. authors: binto george , jayant r. haritsa. venue: the vldb journal – the international journal on very large data bases. year: 2000. Are Product A and Product B the same?" (Answer: No)
   - **Random demo**: "Product A is title: picodbms : scaling down database techniques for the smartcard. authors: patrick valduriez , philippe pucheral , christophe bobineau , luc bouganim. venue: vldb j.. year: 2001. Product B is title: database techniques for the world-wide web : a survey. authors: daniela florescu , alon levy , alberto mendelzon. venue: acm sigmod record. year: 1998. Are Product A and Product B the same? No"
   - **PSD (Num. clusters: 50)**: "Product A is title: selectivity estimation in spatial databases. authors: viswanath poosala , sridhar ramaswamy , swarup acharya. venue: sigmod conference. year: 1999. Product B is title: effective timestamping in databases. authors: kristian torp , christian s. jensen , richard thomas snodgrass. venue: the vldb journal – the international journal on very large data bases. year: 2000. Are Product A and Product B the same? No"
4. **Walmart-Amazon**
   - **Input**: "Product A is title: sony 16gb class 4 sd memory card. modelno: sf16n4/tqp. Product B is title: pny 4gb class 4 navy sd card. modelno: p-sdhc4g4-ef / navy. Are A and B the Same?" (Answer: No)
   - **Random demo**: "Product A is title: xantech cpl10 rf ir coupler. modelno: cpl10. Product B is title: xantech cpl10 rf ir coupler. modelno: nan. Are A and B the Same? Yes"
   - **PSD (Num. clusters: 50)**: "Product A is title: pny 4gb sdhc memory card class 4. modelno: p-sdhc4g4-ef. Product B is title: pny optima 2gb sd class 4 flash memory card p-sd2gb-ef. modelno: p-sd2gb-ef. Are A and B the Same? No"
5. **Adult**
   - **Input**: "age: 18-21. workclass: Private. education: Some-college. maritalstatus: Never-married. occupation: Craft-repair. relationship: Own-child. race: White. sex: Male. hoursperweek: 40. country: United-States. income: LesshTan50K. Is there an error in age: 18-21?" (Answer: No)
   - **Random demo**: "age: 31-50. workclass: Private. education: Some-college. maritalstatus: Married-civ-spouse. occupation: Prof-specialty. relationship: Wife. race: White. sex: Female. hoursperweek: 36. country: United-States. income: MoreThan50K. Is there an error in relationship: Wife? No"

- **PSD (Num. clusters: 50)**: "age: 18-21. workclass: Private. education: Some-college. maritalstatus: Never-married. occupation: Other-service. relationship: Own-child. race: White. sex: Male. hoursperweek: 18-21. country: United-States. income: LesThan50K"

6. **Hospital**
   - **Input**: "Is there a x spelling error in EmergencyService: yes?" (Answer: No)
   - **Random demo**: "Is there a x spelling error in Address1: 101 hospital circle? No"
   - **PSD (Num. clusters: 50)**: "Is there a x spelling error in EmergencyService: yes? No"