Matching Table Metadata with Business Glossaries Using Large Language Models

Elita Lobo[2], Oktie Hassanzadeh[1], Nhan Pham[1], Nandana Mihindukulasooriya[1], Dharmashankar Subramanian[1] and Horst Samulowitz[1]

[1]IBM Research, Yorktown Heights, NY, United States
[2]University of Massachusetts Amherst, MA, United States

Abstract

Enterprises often own large collections of structured data in the form of large databases or an enterprise data lake. Such data collections come with limited metadata and strict access policies that could limit access to the data contents and, therefore, limit the application of classic retrieval and analysis solutions. As a result, there is a need for solutions that can effectively utilize the available metadata. In this paper, we study the problem of matching table metadata to a business glossary containing data labels and descriptions. The resulting matching enables the use of an available or curated business glossary for retrieval and analysis without or before requesting access to the data contents. One solution to this problem is to use manually-defined rules or similarity measures on column names and glossary descriptions (or their vector embeddings) to find the closest match. However, such approaches need to be tuned through manual labeling and cannot handle many business glossaries that contain a combination of simple as well as complex and long descriptions. In this work, we leverage the power of large language models (LLMs) to design generic matching methods that do not require manual tuning and can identify complex relations between column names and glossaries. We propose methods that utilize LLMs in two ways: a) by generating additional context for column names that can aid with matching b) by using LLMs to directly infer if there is a relation between column names and glossary descriptions. Our preliminary experimental results show the effectiveness of our proposed methods.

1. Introduction

Large collections of structured tabular data that businesses possess can be invaluable resources for various analytic tasks. Traditionally, such data collections are gathered in large databases or data warehouses, along with mechanisms of collecting and maintaining metadata with well-curated schemas, data catalogs, or master data as a part of a master data management solution. In practice, the overhead of maintaining accurate metadata may be prohibitively difficult and expensive. More recently, enterprises are moving toward collecting all their data in data lakes without any requirements or strict enforcement of metadata availability or quality.

OM-2023: The 18th International Workshop on Ontology Matching, November 7th, 2023, Athens, Greece
[†]Work done while at IBM Research.
loboelita@gmail.com (E. Lobo); hassanzadeh@us.ibm.com (O. Hassanzadeh); nhp@ibm.com (N. Pham); nandana@ibm.com (N. Mihindukulasooriya); dharmash@us.ibm.com (D. Subramanian); samulowitz@us.ibm.com (H. Samulowitz)
0000-0001-5307-9857 (O. Hassanzadeh); 0000-0003-1707-4842 (N. Mihindukulasooriya); 0000-0002-1990-7740 (D. Subramanian); 0000-0002-6780-3217 (H. Samulowitz)

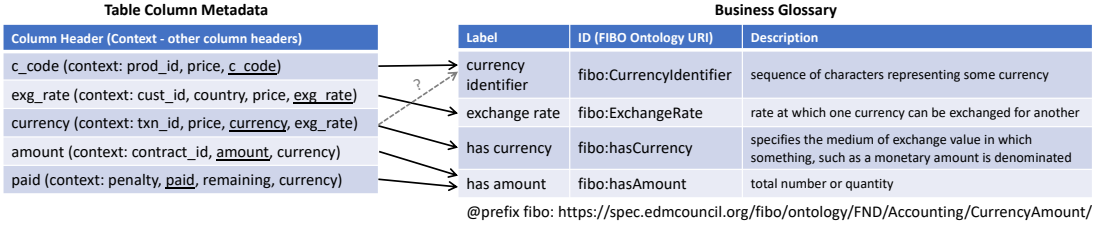

Figure 1: Example column headers and their associated business glossary terms

As a result, there is a need for solutions that can effectively use limited metadata, such as column headers, and automatically generate useful metadata. Most organizations maintain some business glossary [1] with a set of concepts that are relevant to the business processes. If table columns can be annotated with business glossary terms, it helps downstream tasks such as data discovery, data integration, or performing advanced analytics.

The task of mapping table columns to a business glossary is similar to the task of annotating a table column with an ontology or knowledge graph concept, which is referred to as the Column Type Annotation (CTA) task [2]. However, to our knowledge, prior work has not considered further restricting the task to using only the table metadata (table name and column headers) and business glossary containing labels and descriptions only. The problem we study in this paper is inspired by our ongoing work on implementing an automated semantic layer for enterprise data lakes [3], and has the following characteristics: 1) we do not have access to data contents due to access restrictions common in enterprise data lakes; 2) we have tabular data with no metadata other than column headers, which is a result of large data imports from highly heterogeneous sources or automated table extraction pipelines; and 3) there is no or very little training data, as the process of manually labeling table columns with business glossary terms is laborious and requires domain expertise. Figure 1 shows a few example column headers along with their context (other column headers in the same table) and their associated business glossary terms.

In the absence of rich metadata and an ontology, the matching process can only rely on the header labels and glossary labels and descriptions. Essentially, the problem becomes a string or text similarity matching problem. Prior work has studied various flavors of such matching methods for record matching in databases [4, 5] as well as ontology alignment [6]. Such methods rely on either *syntactic* matching methods, which rely on common tokens and substrings between the terms that should be matched, or *semantic* matching methods, which rely on the availability of a dictionary of terms along with lists of related terms such as synonyms, hypernyms, and hyponyms. In our setup, we often need to match terms with very little syntactic similarity, and we do not have access to a dictionary that could enable semantic matching. Column headers in tabular data are often cryptic terms, and business glossaries use terminology very specific to a particular enterprise. More recent work has proposed the use of vector representations of terms in the form of embeddings [7]; however, such methods require domain-specific training data and tuning.

In this paper, we propose a novel matching solution that relies on the power of Large Language Models (LLMs) to enable the matching of table columns with glossaries when column headers are not very descriptive and glossary terms do not have a close syntactic similarity to the column headers, and little or no training data is available. In what follows, we first discuss related

work. We then present the problem description, and then we present the details of our solution. In section 5, we present the results of our experiments using real-world enterprise data and business glossaries. We end the paper by outlining a number of lessons learned and avenues for future work.

2. Related Work

A core problem in semantic table understanding [8] is column type annotation, i.e., annotating table columns with a type from an ontology, which enables many business intelligence tasks such as semantic retrieval, data exploration, and knowledge discovery. SemTab challenge [2], which aims at benchmarking systems dealing with the tabular data to KG matching problem, provides several datasets in which the column type annotation task can be evaluated. In the SemTab challenge, this task is formulated as an unsupervised task where participating systems are not given training data.

Column type annotations using table data MTab [9], JenTab [10], and DAGOBAH [11], are example of the systems that participated in the SemTab challenge comprising three KG matching tasks, namely, cell to KG entity (CEA task), column to KG class (CTA task), and column pair to KG property (CPA task). As these systems typically solve the three tasks in a joint manner, they follow a pipeline architecture. The first step links cell mentions to entities within the target ontology. The second step predicts the most likely type for the query column based on the linking results. MTab and DAGOBAH also use additional information from the graph, such as entity relations, to improve cell linking accuracy. It is a requirement for these systems to have cell values that can be mapped to KG entities, which might not be the case in most industry tables.

Column type annotations with only using metadata The problem setup that is studied in this paper differs from the traditional column type annotation task. In our setup, the system that is performing the matching between the table column and glossary concepts will only have access to the table metadata (i.e., table name and column headers) but not the actual table data (i.e., cell values). This problem setup has similarities with the ontology matching methods that rely purely on string similarity measures. String similarity measures have been studied extensively for various matching tasks, including in ontology alignment [6]. Syntactic measures of similarity measure how close two strings are based on measuring the overlap between tokens or substrings in two strings or measures based on the number of character edit operations that can transform one string to another. Examples of such methods are edit similarity and Jaro-Winkler [12]. While such approaches have shown very promising performance in various matching tasks, they are inherently not capable of differentiating between strings that are syntactically very similar but semantically dissimilar. Classic semantic measures rely on resources such as WordNet [13] containing related terms. The application of those methods is limited to when such resources are available. More recently, methods that rely on vector representations for semantic similarity have shown superior performance in various tasks. Initial approaches relied on word2vec [14], which can handle semantic matching between words

and short phrases. More recently, word embeddings [15, 16] and sentence embeddings [7] have shown promising performance in semantic textual similarity tasks. As we will show in our experiments, business glossaries often have very similar labels and descriptions that these sentence transformer-based approaches alone cannot effectively differentiate between.

3. Preliminaries

3.1. Problem Setup

We assume a setting in which we have only superficial tabular metadata corresponding to any chosen table in the form of a list of n superficial metadata fields, say $M = \{(M_i)_{i=1}^{n}\}$. In practice, this list could contain a main column name of interest that must be matched with the correct business concept in a glossary, along with the other column names in the table under question. We also assume access to a business glossary that consists of m glossary items $\mathcal{G} = \{(l_j, d_j)\}_{j=1}^{m}$. Here, l_j and $d_j \forall j \in [1, m]$ represent respectively the *label* and *description* of the j^{th} glossary item. For example, the glossary could be a list of tuples containing labels and descriptions of various business concepts. Given such superficial metadata M, the task is to find its closest glossary item match.

In this paper, we consider a relaxed version of the glossary matching problem, where the task is to select k glossary items for any given metadata M such that it maximizes the probability of **Hit@k** for any given M. A **Hit@k** for a given metadata represents a Boolean variable that takes the value *one* if the selected k glossary items contain the closest match of the metadata and *zero* otherwise. Finally, we also assume that we have a human feedback bank available in the form of l tuples $\mathcal{H} = \{(M_k, G_k)_{k=1}^{l}\}$ where M_k represents some metadata and $G_k \in \mathcal{G}$ represents the correct glossary match. We will use the human feedback bank $\mathcal{H}$ to construct task demonstrations for the In-Context Learning approach described in the next section.

3.2. Large Language Models

Recent work [17, 18, 19, 20, 21] has demonstrated that Large Language Models (LLMs) perform extraordinarily well on instruction-based tasks as long as these tasks can be represented in natural language. LLMs are transformer models with billions of parameters, trained on large data corpora and fine-tuned on instructions-based tasks, including classification tasks, generation tasks, and question-answering tasks. An LLM takes as input a prompt containing the description of the task along with additional context represented using natural language and outputs the results of the task in natural language. In this work, we leverage LLMs, specifically, Flan-t5-models [22] to obtain more accurate metadata for business glossary matching. Since LLMs have been trained on large data corpora, they can identify complex relations and patterns between different objects in natural language and, thereby, can be used to obtain more accurate matching.

3.3. In-Context Learning

Fine-tuning LLMs for new tasks or datasets is often very computationally expensive and requires large amounts of data, which is often not feasible. A common approach to evade this problem

question: column name: customer numeric. , other column names: customer id , full name , customer status , customer status product , customer status reason , description , basic data incomplete , residence area id. , column description:
answer: specifies the unique identification of the customer. [SEP]
question: column name: communication status. , other column names: cost amount , employee id , customer id , communication id , planned end date , planned start date , actual start date , actual end date , communications version anchor. , column description:
answer: a term that distinguishes between communications according to the status of the lifecycle; for example, a communication may be awaiting further information, closed or open. [SEP]
question: column name: customer status. , other column names: industry code , customer tenure range , customer active tenure range , staff turnover range , legal form , staffing structure , purpose , franchised , customer status tenure range , organization customer profile id , customer market segment , location country , introduction. , column description:
answer:

Figure 2: Example prompt for tuned and pre-tuned Flan-t5 models used in MDG-MICL (2-shots) method.

is by appending one (one-shot) or multiple (multi-shot) demonstrations of the task in natural language to the prompt. This is commonly known as the One-shot or Multi-shot In-Context Learning (ICL) or In-Context Prompting method [23]. Figure 2 shows an example of multi-shot in-context learning on a classification task. In-context learning is known to have worked well for several new problems in the prior literature. This work uses the human feedback bank to generate relevant demonstrations for in-context learning. We conjecture that ICL, with good demonstrations, can improve the performance of the glossary matching task without additional fine-tuning.

4. Methodology

We propose two different classes of methods a) Metadata Description to Glossary Matching (MDGM) b) Direct Metadata to Glossary Matching (DMGM) that retrieve a set of k glossary items for any given metadata such that it contains the glossary match with high probability. In MDGM methods, we use the Large Language Models to obtain a metadata description and use the description to retrieve k glossary items that are most similar to the given description in some latent space. On the other hand, in the DMGM methods, we use LLMs to directly match metadata to business glossaries. More specifically, we treat the metadata to glossary matching problem as either a Boolean classification or a Multi-class classification problem. We design special prompts to LLMs to *directly* infer which glossary items are potential descriptions of the metadata and choose the top-k glossary items most likely to be the description of the given metadata. Although MDGM methods seem like an indirect approach to glossary matching, they can be useful when the glossary constantly changes during test time, and direct inference over large glossaries is expensive. We show in section 5.2 that MDGM methods tend to outperform DMGM methods.

4.1. Column Description to Glossary Matching

We now describe various techniques we propose for generating descriptions of metadata for the metadata to business glossary matching problem.

4.1.1. Metadata Description Generation via Multi-Shot In-Context Learning (MDG-MICL)

Since LLMs are trained on large data corpora, they can generate good descriptions of any concept they may have seen during the training period. In this method, we leverage this

We have a database table with columns: industry code, customer tenure range, customer active tenure range, staff turnover range, legal form, staffing structure, purpose, franchised, customer status tenure range, customer status, organization customer profile id, customer market segment, customer profitability segment, revenue range, profit after tax range, amount owed range, age range, preferred payment method, debit credit status, spoken language, location country, introduction.
Answer this question, making sure that the answer is supposed by the text: Is "A term that distinguishes between Involved Party / Location Relationships according to the specific importance of the address in relation to conducting business with Involved Parties." a description of "importance level"?

yes,no

Figure 3: Example prompt for tuned Flan-T5-XL and pre-tuned Flan-T5-XXL models used in MDG-Cl method.

knowledge of LLM to generate descriptions of the given metadata. We construct a special prompt instructing the LLM to generate a metadata description. Further, we use ICL to improve the quality and control the format of the description generated by the LLM. Figure 2 shows an example of the ICL prompt used for this task with tuned Flan-T5-XL and Flan-T5-XXL models. To construct demonstrations for ICL, we proceed as follows. Using the Sentence-BERT (SBERT) sentence embeddings [7], we first generate sentence embeddings of the metadata and all the descriptions corresponding to the glossary items in the human-feedback bank, $\mathscr{H}$. We use the cosine similarity metric to find e glossary item descriptions from $\mathscr{H}$ closest to the metadata in the SBERT sentence embedding space. We construct demonstrations from these e glossary items in $\mathscr{H}$, and append them to the prompt, in response to which the LLM generates a description. We obtain the final description by appending the table name and metadata to the LLM-generated description. We embed this description using SBERT and obtain the top-k glossary items by computing the cosine similarity metric between this embedding and the sentence embeddings of all the glossary item descriptions. The procedure of computing top-k glossary items from the glossary set $\mathscr{G}$ for a given metadata description using the SBERT sentence embeddings and cosine similarity metric is used in several subsequent methods; for the sake of brevity, we will here on refer to this procedure as the SBERT $k-$nearest neighbors method.

4.1.2. Metadata Description Generation via Classification (MDG-Cl)

As discussed in section 3, LLMs can also identify complex relations between various concepts in natural language. Therefore, in this method, we leverage LLMs to directly select the best description from the set of glossary descriptions in $\mathscr{G}$ using a classification-based technique.

Specifically, for a given metadata and glossary set $\mathscr{G}$, we construct a binary classification prompt against each glossary item in the set $\mathscr{G}$, that queries the LLM on whether the given glossary item is a potential description of the metadata. Figure 2 shows an example of the classification prompt used for this task. Note that when the glossary set $\mathscr{G}$ is too large, it may result in high inference costs. We can mitigate these high costs by first shortlisting the top $k_1, (k_1 > k)$ glossary item from the glossary set $\mathscr{G}$ using the SBERT $k-$nearest neighbors method before proceeding with the classification prompts.

The final description of the metadata corresponds to the glossary description for which the classification response was positive with the highest log probability score of the classification task, appended to the table name and the metadata itself. If no such glossary item exists, we use the metadata itself as the description. Once the metadata description is generated, we select the top-k glossary items from the glossary set $\mathscr{G}$ using SBERT $k-$nearest neighbors described in MDG-MICL method.

The main column name in the table is "customer status" and the other column names are industry code, customer tenure range, customer active tenure range, residence area id.

Based on the above text, what is the best description of "customer status" from the following choices?

1. Customer Performance Status : A term that distinguishes between Customers according to the degree to which the Customer is judged to be performing or non-performing. The formula for deriving this judgment will be client-specific, but factors which could be taken into account include: the Customer Credit Risk Rating, the Customer Non-Performing Loan Status, the balance of the Customer's non-performing loans compared to the Customer Funds Under Management, the various risk-related classifications of the Customer's Finance Service Arrangements, etc.
2. Customer Relationship Life Cycle Status : A term that distinguishes between Customers according to the specific life cycle state in which the Customer relationship exists.
3. Customer Relationship Review : Identifies a Status Review for the purposes of meeting with the Customer to enhance the customer relationship.
4. Status Review : Identifies a Business Activity in which the status of an item is reviewed to determine if it is still valid; for example; confirmation of bankruptcy status of an Individual.
5. None of the above

Figure 4: Prompt for Flan-T5-XL and pre-tuned Flan-T5-XXL models used in MDG-MCQA method.

4.1.3. Metadata Description Generation via Multiple Choice Question Answering (MDG-MCQA)

An alternative way of generating descriptions using LLMs is by using a Multiple Choice Question Answer (MCQA) prompt, as shown in Figure 4, that instructs the LLM to choose the best description of the metadata amongst the descriptions of selected glossary items. Although this may seem counterintuitive, we observe in our experiments that using the description of the selected glossary item to find the top-k glossary items instead of simply returning the glossary item corresponding to the selected description results in a higher Hit@5 rate. Similar to previous methods, we shortlist the top $k_1 (k_1 > k)$ glossary items that are closest to the metadata in sentence-embedding space and use them as choices in our Multiple Choice Question Answer prompt. We also add "None of the above" to the list of choices in the MCQA prompt. Finally, we append the metadata and the table name to the description of the LLM-selected glossary item and use it to find the top-k glossary items using the SBERT $k-$nearest neighbors method. Note that when the LLM selects the "None of the above" option, we use the metadata itself as the description.

4.2. Direct Metadata to Glossary Matching

4.2.1. Direct Inference via Classification (DI-Cl)

This method is a variant of the MDG-Cl method that uses LLM to directly select the top-k glossary items for any given metadata without generating a description of the metadata. Similar to previous methods, we shortlist the top k_1 $(k_1 > k)$ glossary items closest to the metadata in the sentence-embedding space. For each of these glossary items, we construct binary classification prompts that query the LLM on whether the description of the glossary item matches the metadata. An example prompt is shown in Figure 5. Among all glossary items with positive responses, the glossary items with top-k highest log probability scores are selected. It is important to note that this method may return less than k glossary items. Such missing items are assumed to be incorrect matches while computing the **Hit@k** metric.

4.2.2. Direct Inference via Multiple Choice Question Answering (DI-MCQA)

We consider another variation of the MDG-MCQA method that computes a single best match of the given metadata without generating a description of the metadata. In this method, we

Is "The text value of the Alternative Identifier of the Communication." a description of the column name: "importance level"?
yes,no
Answer: no

Question: Is "The text value of the Alternative Identifier of the Communication." a description of the column name: "communication reference numeric"?
yes,no
Answer: yes

Question: Is "Identifies a Hierarchy Level that is on the bottom of the hierarchy." a description of the column name: "importance level"?
yes,no
Answer:

Figure 5: Example prompt for all pre-tuned and tuned Flan-t5 models used in DI-Cl method.

We have a database table with columns: industry code, customer tenure range, customer active tenure range, staff turnover range, legal form, staffing structure, purpose, franchised, customer status tenure range.

What is the correct description of "importance level"?

Choose the best answer to the above question from the following choices:

1. Customer Importance Rating : Identifies a Rating Scale that represents the degree to which a customer should be given special treatment.
2. Priority Rating : Identifies a Rating Scale that is used to represent the priority or importance of one object over another one; for example, the bank's vaults are more important that company chairs (RI priority), serving a VIP customer is more important than a daily status meeting, etc.
3. Involved Party Hierarchy Level : Indicates the level of the Involved Party in the Hierarchy.
4. Education Level Criterion : Identifies an Individual Criterion according to the amount of formal training attained by the individual.
5. None of the above

Figure 6: Example prompt for Flan-T5-XL and pre-tuned Flan-T5-XXL models used in DI-MCQA method.

follow the same procedure as MDG-MCQA to construct Multi-Choice Classification prompts, as shown in Figure 6, and return the glossary item corresponding to the description selected by the LLM. Since this method always returns a single glossary item, we will assume that the $k-1$ missing glossary items are incorrect matches while computing the **Hit@k** metric.

5. Experiments

In this section, we empirically evaluate and compare the performance of all the MDGM and DMGM methods proposed in section 4. Specifically, we investigate the following two questions a) Which of the proposed methods: MDG-MICL (0-shot, 1-shot, 2-shots), MDG-Cl, MDG-MCQA, DI-Cl, and DI-MCQA, best leverage LLM in solving the metadata to glossary matching task, and b) is it possible to obtain more accurate matching, i.e., higher **Hit@5** and **Hit@1** using LLMs than using basic similarity-score based matching methods? Our preliminary results show that LLMs are indeed effective in improving glossary matching accuracy.

5.1. Experimental setup

In all our experiments, the metadata consists of the column name of interest and the other column names in the table. The glossary is a list of tuples where each tuple consists of a label and a description of the label. Multiple column names may be matched to the same label. Furthermore, the label of the matched glossary item for each column name may not coincide with the column name. We evaluate our methods on the Flan-T5-XL, Flan-T5-XXL [24, 22], and a Flan-T5-XL model that we fine-tune on the training dataset using the supervised fine-tuning method for LLMs [25]. For each method and LLM model, we experiment with 4-6 different

prompt templates. However, due to lack of space, we only provide examples of the prompt templates that achieved the highest hit@5 rate (2,3,4,5,6).

We use the `all-mpnet-base-v2` SBERT model from the `sentence-transformers` library[7] in all experiments. We evaluate each method based on their **Hit@5** and **Hit@1** rates on the test dataset, i.e., the empirical mean of **Hit@5** and **Hit@1** computed on the test dataset. These measures were chosen to reflect our goal of having the correct glossary item as the top or within the top 5 glossary items returned to the user on a GUI [3]. We compare these scores against ones produced by a baseline, which computes the top-k glossary items based on the cosine similarity of the sentence embedding between the column name and the descriptions in the glossary.

MDE Dataset The MDE Dataset is an IBM-internal benchmark developed by annotating the column names of the "Customer Insight" example database of "IBM InfoSphere Warehouse Pack" [26] with theevaluateglossary terms from the IBM Knowledge to fine-tune Financial Services (IBM KAFS) [27] glossary. The Customer Insight database consists of 26 tables with 688 columns. The column names contain cryptic codes and abbreviations to reflect realistic tables commonly seen in client engagements. The IBM KAFS glossary contains 9,137 business terms with their labels and descriptions. Out of 688 columns, 488 have suitable matching terms in the glossary, and the rest are annotated as null mappings and ignored in the evaluation. We split the mappings into train, test, and demonstration splits with 208, 212, and 68 columns, respectively. These splits contain tuples of the form (*column_name*, *other_column_names*, *glossary_item*). The training, test, and demonstration splits are used to fine-tune the LLM model, evaluate the method, and as a proxy for human feedback.

5.2. Experimental Results

Table 1 shows the **Hit@5** and **Hit@1** rates achieved by different methods on the MDE dataset with different LLM models. As expected, the **Hit@5** rate increases with the number of demonstrations in the MDG-MICL method. This result suggests that it may be beneficial to use in-context learning whenever demonstrations are readily available. However, we do not observe a similar trend for *Hit@1* rate. We believe this is due to the difficulty of matching metadata to a single glossary item when multiple glossary items have similar descriptions. Overall, MDG-MICL achieves the highest Hit@5 and **Hit@1** scores and significantly outperforms the baseline method when two demonstrations are provided in the prompt. Meanwhile, we observe that the DI-Cl and DI-MCQA methods achieve the worst **Hit@5** rates. We conjecture that this may be due to the underlying biases of LLMs towards certain class labels in classification and question-answering tasks as observed in prior works [28, 29]. These biases can be corrected using various calibration techniques [28, 29], which we leave for future work. It is also important to note that the DI-MCQA method selects a single best glossary match and, thus, more likely fails to select the correct item when multiple glossary items have similar descriptions. These preliminary results indicate that LLMs alone may not improve the matching accuracy. Finally, although MDG-Cl and MDG-MCQA use the same classification prompt and Multiple Choice Question Answer prompts as DI-Cl and DI-MCQA, they achieve higher **Hit@5** and **Hit@1**

Methods	Flan-T5-XL		Flan-T5-XXL		Tuned Flan-T5-XL	
	Hit@1	**Hit@5**	**Hit@1**	**Hit@5**	**Hit@1**	**Hit@5**
Baseline Method	0.4575	0.7406	0.4575	0.7406	0.4575	0.7406
MDG-MICL (0-shot)	**0.5**	**0.7877**	**0.4858**	**0.7689**	0.4434	**0.7594**
MDG-MICL (1-shot)	**0.4764**	**0.7972**	**0.4953**	**0.8113**	0.467	**0.75**
MDG-MICL (2-shots)	**0.5047**	**0.8133**	**0.4858**	**0.816**	**0.5**	**0.7877**
MDG-Cl	0.434	**0.7453**	**0.5**	**0.7642**	0.4575	**0.7547**
MDG-MCQA	**0.5708**	**0.7547**	**0.5755**	**0.7453**	**0.5753**	**0.7642**
DI-Cl	0.3585	0.4717	**0.4811**	0.6934	0.3632	0.4953
DI-MCQA	**0.5519**	0.5519	**0.5708**	0.5708	**0.5519**	0.5519

Table 1
This table shows the **Hit@5** and **Hit@1** scores achieved by the Baseline, MDG-MICL (0,1,2)-shots, MDG-Cl, MDG-MCQA, DI-Cl and DI-MCQA methods on Flan-T5-XL, Flan-T5-XXL and Tuned Flan-T5-XL models. The highlighted numbers indicate that the corresponding methods have outperformed the baseline method in terms of the **Hit@5**/**Hit@1** value. MDG-MICL consistently achieves the highest **Hit@5** and **Hit@1**, whereas DI-Cl and DI-MCQA have the worst **Hit@5**.

than the latter methods. We believe that this is because these methods select glossary items whose descriptions are similar to the closest glossary match and, thus, tend to perform better.

6. Discussion and Future work

This paper proposes two different classes of methods, i.e., MDGM and DMGM, that leverage LLMs for solving the metadata to glossary matching problem. MDGM methods use LLMs to generate good metadata descriptions, which we couple with similarity-based metrics for more refined matches. This class of methods is necessary when the glossary is likely to change during test time frequently, and repeated inferences are expensive. The second class of methods (DMGM) utilizes LLMs to infer which glossary items are potential matches directly. These methods are helpful when the metadata is too complex and there is a significant difference between the description of the metadata and that of its closest glossary match. Although we have shown that many of these methods can potentially obtain more accurate glossary matching, we can further improve them in several ways. Our experiments show that DMGM methods perform poorly compared to MDGM methods. This may be due to the undesirable biases towards specific class labels that LLMs learned during training. One approach to mitigating these biases is using various calibration techniques [28, 29]. Providing several positive and negative demonstrations in the classification and multiple-choice question-answer prompts may also help mitigate LLMs' default biases [23]. We can further improve the MDGM methods that generate descriptions of metadata by constraining LLMs to sample words mainly from the glossary or providing LLMs with the top-k glossary items and prompting LLMs to generate descriptions similar to those of the glossary items. This can be achieved by using the Constrained Beam Search algorithm [30] or simply manipulating the output distribution of LLMs before sampling such that it assigns higher weights to words from the glossary. It may also be helpful to use various prompt-tuning and prompt-editing methods [31] to further improve the efficiency of the prompts used with

LLMs. Although these directions remain intriguing, they warrant more in-depth empirical study, which we leave for future work.

References

[1] P. Sarkar, Business Glossary for DaaS, 2015, pp. 103–122.

[2] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas, Semtab 2019: Resources to benchmark tabular data to knowledge graph matching systems, in: European Semantic Web Conference, Springer, 2020, pp. 514–530.

[3] D. K. I. Weidele, G. Rossiello, G. Bramble, A. N. Valente, S. Bagchi, M. F. M. Chowdhury, M. Martino, N. Mihindukulasooriya, H. Ananthakrishnan, H. Strobelt, H. Patel, A. M. Gliozzo, O. Cornec, A. Gupta, S. Mehta, M. Kesarwani, H. Samulowitz, O. Hassanzadeh, A. Agarwal, L. Amini, Conversational GUI for Semantic Automation Layer, 2023. Under Submission to ISWC 2023, Industry Track (will be on arxiv soon).

[4] A. Chandel, O. Hassanzadeh, N. Koudas, M. Sadoghi, D. Srivastava, Benchmarking declarative approximate selection predicates, in: SIGMOD, 2007, pp. 353–364.

[5] O. Hassanzadeh, M. Sadoghi, R. J. Miller, Accuracy of approximate string joins using grams, in: QDB, 2007.

[6] M. Cheatham, P. Hitzler, String similarity metrics for ontology alignment, in: The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II, volume 8219, Springer, 2013, pp. 294–309.

[7] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019.

[8] J. Pujara, P. Szekely, H. Sun, M. Chen, From tables to knowledge: Recent advances in table understanding, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 4060–4061.

[9] P. Nguyen, I. Yamada, N. Kertkeidkachorn, R. Ichise, H. Takeda, Semtab 2021: Tabular data annotation with mtab tool, in: SemTab at (ISWC 2021), 2021, pp. 92–101.

[10] N. Abdelmageed, S. Schindler, JenTab meets semtab 2021's new challenges., in: SemTab@ ISWC, 2021.

[11] V.-P. Huynh, J. Liu, Y. Chabot, F. Deuzé, T. Labbé, P. Monnin, R. Troncy, DAGOBAH: Table and graph contexts for efficient semantic annotation of tabular data, in: SemTab@ ISWC, 2021.

[12] W. E. Winkler, String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage, in: Proceedings of the Section on Survey Research, 1990.

[13] F. Lin, K. Sandkuhl, A survey of exploiting wordnet in ontology matching, in: Artificial Intelligence in Theory and Practice II, Springer US, 2008, pp. 341–350.

[14] J. Liao, Y. Huang, H. Wang, M. Li, Matching ontologies with word2vec model based on cosine similarity, in: Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2021), Springer International Publishing, 2021.

[15] L. L. Wang, C. Bhagavatula, M. Neumann, K. Lo, C. Wilhelm, W. Ammar, Ontology alignment in the biomedical domain using entity definitions and context, in: Proceedings

of the BioNLP 2018 workshop, Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 47–55.
[16] J. Chen, E. Jiménez-Ruiz, I. Horrocks, D. Antonyrajah, A. Hadian, J. Lee, Augmenting ontology alignment by semantic embedding and distant supervision, in: The Semantic Web, Springer International Publishing, Cham, 2021, pp. 392–408.
[17] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. E. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, R. J. Lowe, Training language models to follow instructions with human feedback, ArXiv (2022).
[18] J. Wei, M. P. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, Q. V. Le, Finetuned language models are zero-shot learners, 2022.
[19] Y. Zhou, A. I. Muresanu, Z. Han, K. Paster, S. Pitis, H. Chan, J. Ba, Large language models are human-level prompt engineers, in: ICLR, 2023.
[20] J. Jang, S. Ye, M. Seo, Can large language models truly follow your instructions?, in: NeurIPS ML Safety Workshop, 2022.
[21] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J. Nie, J. rong Wen, A survey of large language models, ArXiv abs/2303.18223 (2023).
[22] S. Longpre, L. Hou, T. Vu, A. Webson, H. W. Chung, Y. Tay, D. Zhou, Q. V. Le, B. Zoph, J. Wei, et al., The flan collection: Designing data and methods for effective instruction tuning, arXiv preprint arXiv:2301.13688 (2023).
[23] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, L. Zettlemoyer, Rethinking the role of demonstrations: What makes in-context learning work?, in: EMNLP, 2022.
[24] 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).
[25] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Gray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, R. Lowe, Training language models to follow instructions with human feedback, in: Advances in Neural Information Processing Systems, 2022.
[26] I. U. S. Announcement, Ibm infosphere warehouse pack for customer insight v8.2, 2010.
[27] IBM knowledge accelerator for financial services, 2023.
[28] B. Xu, Q. Wang, Z. Mao, Y. Lyu, Q. She, Y. Zhang, kNN prompting: Beyond-context learning with calibration-free nearest neighbor inference, in: ICLR, 2023.
[29] T. Zhao, M. Wei, J. S. Preston, H. Poon, Llm calibration and automatic hallucination detection via pareto optimal self-supervision, 2023. arXiv:2306.16564.
[30] C. Hokamp, Q. Liu, Lexically constrained decoding for sequence generation using grid beam search, in: ACL, 2017, pp. 1535–1546.
[31] T. Zhang, X. Wang, D. Zhou, D. Schuurmans, J. E. Gonzalez, TEMPERA: Test-time prompt editing via reinforcement learning, in: The Eleventh International Conference on Learning Representations, 2023.