Language Models are Explorers for Join Discovery on Data Lakes

Yaohua Wang * Bolin Ding *† Rong Zhu * Haibin Wang * Zhijian Ma *
Jingren Zhou *

Abstract

Join discovery is a typical task in data lake research. It finds joinable relationships between columns of different tables which is critical for data integration in schema-less data lakes. On the other hand, LLMs have achieved promising results in many natural language tasks recently. They behave well with the in-context learning ability that only needs a few examples and no fine-tuning, showing great advantages that can be applied to join discovery. However, directly applying NLP prompt generation methods to the table modality may lead to performance decline due to differences in data modalities. It means that strategies useful in NLP scenarios cannot be used in data lakes or obtain limited efficacy, leading to sub-optimal prompts. This is because of the differences in the task definition, data modalities and the availability of labeled samples between the two scenarios. Unsuitable prompts fail to harness the full potential of LLMs and may even mislead them into producing incorrect answers. Therefore, a novel **J**oin **D**iscovery **S**ystem with **C**omprehensive and **O**ptimized **P**rompt **E**ngineering (**JD-SCOPE**) is proposed to deal with join discovery on data lakes. It first constructs unsupervised examples for demonstrations of the prompt and the validation set for hyper-parameters tuning. Then it explores the prompt template automatically. Meanwhile, JD-SCOPE also ensures stable outputs of LLMs, thus avoiding the impact of randomness. In addition, JD-SCOPE can be extended to semantic join discovery to explore column pairs that are domain correlated but not identical (e.g., *country code* and *language code*) for a comprehension understanding of data. JD-SCOPE paves a way to ground data lakes with LLMs and harnesses the knowledge and logical reasoning power of LLMs for join discovery. To evaluate join discovery effectively, the JD-Lake dataset is curated and benchmarked with baselines and a series of popular LLMs. The experiments prove that JD-SCOPE alleviates this migration problem well and empirically outperforms alternatives, showing the superiority of JD-SCOPE.

Keywords: Data Lakes, LLMs, Join Discovery

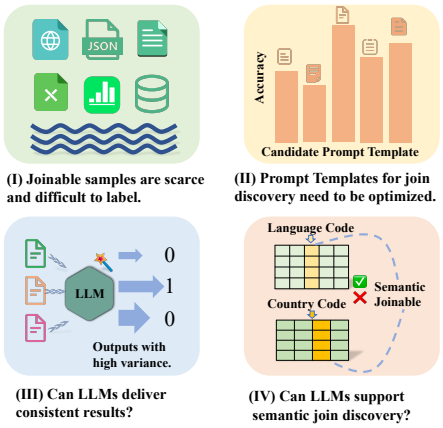

Figure 1: LLM challenges in data lake join discovery.

1 Introduction

1.1 Background and Motivation The data lake [26, 29, 32] is a modern approach to enterprise data architecture that has attracted a great deal of relevant research [13, 17, 18, 22, 25, 30, 33, 35, 38]. One meaningful direction among them is the join discovery [10, 14, 15, 19, 20, 40, 49]. It finds joinable relationships between tables in this schema-less repository where relationships(e.g., PK/FK relationships) between tables are lost. These restored relationships can prevent a data lake from degrading into a data swamp, serve as a fundamental block in database-related fields (e.g., data integration) as well as machine learning fields (e.g., feature augmentation).

Recently efforts on join discovery can be roughly divided into overlap-based, embedding-based and learning-based methods. The overlap-based methods [14, 40, 49] count the intersection size of two candidate columns to determine whether they can be joined. However, the column headers and table captions are not well utilized which may convey critical semantic knowledge. In addition, this statistical approach is sensitive to the skewed data distribution within columns and tends to yield false negative cases, meaning joinable columns are misdiagnosed as dis-joinable because of the low overlap. The embedding-based methods [10, 15] capture similarity relationships of columns according to

*Alibaba Group, Email: {xiachen.wyh, bolin.ding, red.zr, binke.whb, zhijian.mzj, jingren.zhou}@alibaba-inc.com
†The corresponding author.

the column embeddings by pre-trained models [11, 21]. Nevertheless, they are generic solutions to join discovery and lack flexibility in terms of specialized domains or adding new context. As a result, they are prone to producing sub-optimal results for the challenging join discovery. The learning-based methods [19, 20] rely on labeled samples to train classifiers, facing the problem of generalizing to different scenarios, and is therefore impractical on the data lake scenario. Thus, the join discovery realm is seeking a technological revolution.

On the other side, the NLP landscape has been reshaped by large language models (LLMs) [7, 11, 34, 36]. LLMs are deep neural network models trained on massive amounts of unlabeled text with many GPUs. The backbone of LLMs is usually Transformers with the well-known self-attention mechanism [39]. Recently, it has been found that LLMs demonstrate amazing and versatile emergent ability [42, 45] when scaling up their model size and training set. With the technologies above, LLMs can learn the language rules automatically and achieve state-of-the-art results on many NLP benchmarks even with few labeled samples. This is attributed to the fact that LLMs can store vast amounts of factual knowledge, conduct certain logical reasoning, generalize, and summarize. Therefore, LLMs are also promising for join discovery.

1.2 Challenges and Our Contributions The performance of LLMs relies heavily on the design of prompts [7]. A minor modification in the prompt can result in a significant alteration in the output of LLMs [27]. The prompt is a flexible text framework containing the necessary information and questions and is used to query answers from LLMs. Prompts are well-studied in the NLP field [7, 8, 27, 43, 46, 47]. However, directly applying related prompts generation methods [7, 47] on join discovery in the data lake scenario may encounter the migration problem. It means that strategies useful in NLP scenarios cannot be used in data lakes or obtain limited efficacy. This is because of the differences in the task definition, data modalities and the availability of labeled samples between the two scenarios: Join discovery is a business problem that can be modeled from multiple perspectives, which is specific to databases and differs from tasks in NLP; The tabular modal data in data lakes is significantly different from the natural language used in the training phase, which is arranged with rows and columns rather than easily understandable sentences; Labeled samples in data lakes are not as abundant as those in the NLP field. All these differences contribute to the migration problem, which in turn leads to sub-optimal prompts. The unsuitable prompts not only fail to inspire superior abilities of LLMs but even mislead LLMs to output incorrect results, leading to performance degradation. This problem cannot be ignored as tabular data is common in data lakes but has little study interest.

The challenges to handling this migration issue are four-folds. First, it is difficult to obtain labeled column pairs. As presented in Fig. 1 (I), the source of data lakes is intricate, spanning web, JSON files, logs, tables, charts and databases. These data are independent of each other. Labeling them is a tedious task as it requires domain knowledge of these data sources, which is often incomplete in data lakes, further increasing the difficulty of annotation. Without labeled samples, the power of in-context learning cannot be realized [7], which in turn leads to sub-optimal results. Second, it is tricky to find a suitable prompt template for join discovery in data lakes, which is critical to results as shown in Fig. 1 (II). For example, the instruction of the prompt template affects the LLM's understanding of join discovery. The describing form of join discovery (denoted by *task form* in this paper) affects LLM's thinking on join discovery tasks. The task form that best fits the LLMs can yield the desired results. Third, the outputs of LLMs may have a high variance as shown in Fig. 1 (III). There is randomness in the sampling of demonstrations and cell values when constructing prompts. There is also randomness in token selection when LLMs generate the results. All these contribute to a high variance in outputs. Therefore, *can LLMs deliver stable and consistent results for join discovery tasks?* Last, semantic join discovery is required on data lakes as depicted in Fig. 1 (IV). Analysis on data lakes tends to be ad-hoc and in-situ without any ETL operations, and no "bridge table" [23] with desired mapping between two candidate columns in data lakes. Advanced analysis needs to consider column pairs that are conceptually correlated but not identical (*country code* and *language code*) for a comprehension understanding of data. Thus, *can LLMs support this kind of semantic join discovery*?

To overcome these tough challenges, an *Unsupervised Example Construction* strategy is first proposed to address the lack of labeled examples problem. This strategy takes advantage of the tabular dataset, constructing positive and negative samples by sampling within and between columns. To further refine the constructed examples, the corresponding metadata is also generated by LLMs. The constructed examples can serve as both the demonstrations in prompts and the validation set for hyper-parameters tuning. Next, an *Auto Template Exploration* module is put forward to find the suitable prompt template for join discovery by searching and evaluating alternatively. The search space includes both the instruction and task form, where

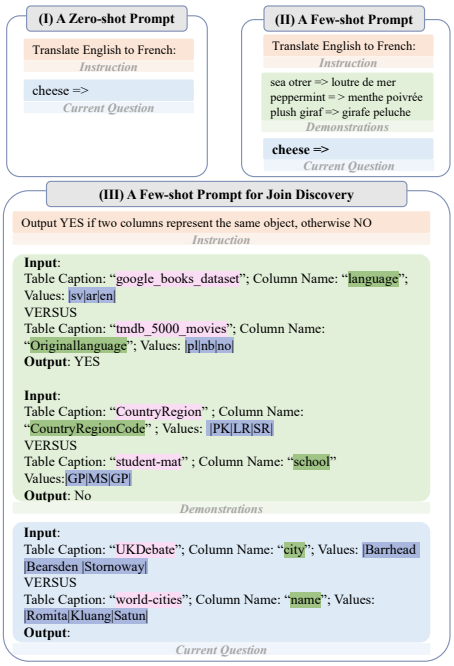

Figure 2: Example prompts on NLP and join discovery.

the candidate instructions are composed by LLMs and task forms are predefined. Then, an *Ouput Result Stabilization* module is presented to enable LLMs to output stable results inspired by the bootstrapping [16]. Finally, the *Semantic Join Discovery Adaption* strategy is introduced to allow LLMs to support the semantic join discovery by retrieving similar examples and using them for demonstrations. Based on these ideas, a novel **J**oin **D**iscovery **S**ystem with **C**omprehensive and **O**ptimized **P**rompt **E**ngineering (**JD-SCOPE**) is proposed to solve the migration problem and enable LLMs an explorer for join discovery on data lakes. The main contributions are summarized as follows: (1) This paper presents the migration problem when using LLMs for join discovery in data lakes, and identifies four challenges in addressing it, which is meaningful and insightful. (2) This paper outlines the next generation of join discovery architecture with LLMs, achieving deep integration of both fields and producing a new tool with a high potential for future impact, which is promising and pioneering. (3) This paper proposes the JD-SCOPE system, which addresses the challenge of lacking labeled examples, finds optimal prompt templates, produces stable results and is extended to semantic join discovery, which is effective and efficient. (4) This paper conducts extensive experiments to verify the superiority of JD-SCOPE and effectiveness of each module, benchmarks JD-SCOPE on a series of popular LLMs and curates a manually annotated dataset named JD-Lake for join discovery on data lakes, which is solid and complete.

2 Methodology

2.1 Workflow of JD-SCOPE Prompts are usually written carefully by humans according to some principles as show in Fig. 2 (I and II). It can be *imitated* on the join discovery task in Fig. 2 (III). However, the *imitated* version prompt in Fig. 2 (III) does not yield satisfactory results as discussed in Sec. 3.4.2. The causes are manifold and can be analyzed from each part of the prompt. JD-SCOPE is proposed to improve that *imitated* version prompt and each of its modules. The **construction and evaluation of prompts** for LLMs is illustrated in Fig. 3. The input to the JD-SCOPE is the data lake which contains many independent data tables from different sources. JD-SCOPE looks for prompt templates suitable for join discovery based on these tables. The instruction and the task form to describe the join discovery are the focus of the search. After the search is complete, the optimal prompt template with a specific instruction and task form is produced. Candidate column pairs from data lakes can wear this optimal prompt template and are then fed into LLMs to obtain the final prediction. However, key issues need to be further addressed to make JD-SCOPE work. ***How*** *can we construct effective demonstrations of prompts on unlabeled data lakes?* ***How*** *can we efficiently search for instructions and task forms for join discovery, and then evaluate the candidates in a self-driven way?* ***How*** *can we ensure stable and consistent output of LLMs?* ***How*** *can we extend the application of JD-SCOPE to more advanced semantic join discovery tasks?* JD-SCOPE proposes corresponding components to handle these challenges as follows.

2.2 Unsupervised Examples Construction

2.2.1 Candidate Column Preparation This process includes two sub-modules for candidate columns.

Candidate Column Selection. Fig. 4 (I) presents the process to construct the positive and negative pairs. One column (displayed in yellow) of data can be sampled twice independently and randomly to get a pair of **twin columns**. As shown in Fig. 2 (III), only a limited number of cell values are needed in one prompt. Random sampling can ensure both identity invariance and diversity, making the twin columns ideally fit the positive pair. On the other hand, any two columns (displayed in orange and deep blue) across the data lake can serve as a pair of negative samples inspired by [15, 28]. These constructed unsupervised examples are composed into the **Demonstration Base** and **Validation Set**, which can be used in the subsequent chapters.

Trivial Pair Elimination. However, the constructed raw examples are not all suitable to act as demon-

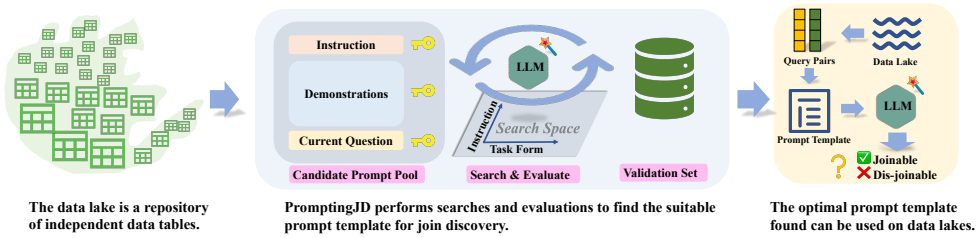

Figure 3: An illustration of the JD-SCOPE. The input is the data lake [25] containing independent data tables. Then JD-SCOPE conducts searches and evaluations alternately in the space of instruction and task form with LLMs. The output is an optimal prompt template that is applied with LLMs to find joinable relationships.

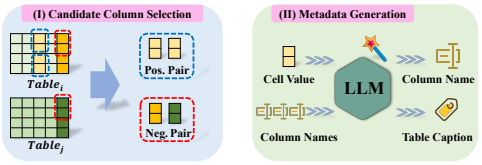

Figure 4: An illustration of the procedure of *Candidate Column Selection* and *Metadata Generation* strategy.

strations. A prompt can only contain limited demonstrations so they need to contain sufficient information to better guide LLMs in making judgments. As a result, two empirical guidelines are suggested when picking negative pairs. First, if two columns have different data types (e.g., *String* v.s. *Integer*), they must be dis-joinable. Second, if the average value length in two columns differs significantly, they are likely dis-joinable.

2.2.2 Metadata Generation The constructed twin columns share **exact the same metadata**, leading to a unnatural and false pattern. This issue is especially fatal in *Candidate Instruction Composition* of Sec. 2.3.2. To deal with this problem, the metadata generation module is proposed to produce a version of metadata so that the metadata between the twin columns makes a difference. It contains two parts, column name generation and table caption generation. When generating the column name, a few cell values are randomly sampled from that column and then integrated into a prompt according to a preset template. Similarly, for each table, the column names are sampled and then constructed into a prompt according to the preset template. The prompt instructs the LLMs to summarize a table caption according to the given column names. The generated version metadata is then used by one of the twin columns randomly. The unnatural and false pattern problem is therefore solved.

2.3 Auto Template Exploration

2.3.1 Task Pool Description This paper proposes six task form instances, and the task pool can be added

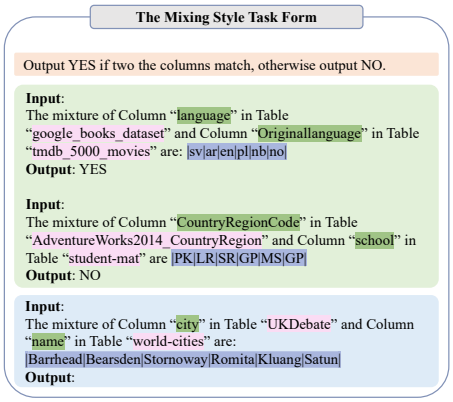

Figure 5: An illustration of the *Mixing Task Form*.

with other forms without affecting the whole framework. The first task form is shown in Fig. 2 (III). The form description is bisected by the word "VERSUS", and each part is represented by a table caption, column name and some values. This form is denoted by **Bisecting Form**. LLMs can directly output judgment results (e.g., "YES" or "NO") according to the form description. The second task form is shown in Fig. 5. Unlike the Bisecting Form, the values of two columns are mixed and this form is denoted by **Mixing Form**. LLMs can also produce judgment results (e.g., "YES" or "NO") according to the Mixing Form description, the same as Bisecting Form. In addition, the table caption and column name can be removed from the above two task forms to see whether these factors have a positive or negative impact on the join discovery. After removing the table caption, **Bisecting W/O T Form** and **Mixing W/O T Form** can be obtained respectively. After removing the column name, **Bisecting W/O C Form** and **Mixing W/O C Form** can be created respectively. Derived task forms are brought together to the form task pool.

2.3.2 Candidate Instruction Composition JD-SCOPE suggests candidates that are then filtered, and the whole process does not require any parameter training. It fully utilizes the comprehension and genera-

tion capabilities of LLMs. First, sample a few examples from the constructed *Demonstration Base* for in-context learning. Second, choose a specific task form from the task pool. Third, build the prompt with the sampled demonstrations wrapped in the specific task form and the predefined prompt template. Then, feed this prompt to LLMs, and obtain the output as candidate instructions. Last, repeat the above process to collect instructions for each task form. After this procedure, each task form is accompanied by a set of instructions. This approach is motivated by the APE in [47].

2.3.3 Search and Evaluation JD-SCOPE searches **both the task form and instruction** jointly, which means the combination of each task form and one of their instructions is the basic unit during the search, instead of instruction alone. The *Validation Set* is constructed by unsupervised examples, as described in Sec. 2.2. The search and evaluated procedure is illustrated in Fig. 3. After obtaining some candidate combinations of task form and instruction, they will be evaluated on the *Validation Set*. The UCB strategy [5] is adopted for efficiency and cost saving. The sampling probability of each unit is according to the UCB scores, which are based on the historical accuracy of the candidate and the number of times it has been reviewed. In the t-th round of search, the UCB assigns score $S_{i,t}$ to unit i. $S_{i,t} = \hat{x}_{i,t} + \sqrt{\frac{2log(t)}{n_{i,t}}}$, where $\hat{x}_{i,t}$ is the average historical accuracy of the unit i in t rounds and $n_{i,t}$ is the number of times the unit i is visited in t rounds. After acquiring a unit, it will be evaluated on a batch of examples from the *Validation Set* to update the $\hat{x}_{i,t}$ and then $S_{i,t}$. After a few rounds of evaluations, the best instruction and task formulation are found according to their estimated score, i.e., $\hat{x}$. Optimized by this module, LLMs can take full advantage of its **logical reasoning** ability and understand the join discovery task.

2.4 Output Result Stabilization LLMs also have probabilistic randomness in outputs. To mitigate this, the multiple tests [6, 9, 24, 41] which is famous for reducing variance in sampling and decision making, are introduced in join discovery. Two candidate columns will be asked multiple times with different values to discriminate whether they are joinable. Thus, the output is then changed into a sequence of "YES" and "NO". The basic version involves asking the same question multiple times with different data fills, while this work defines the improved version.

First, all the current questions and demonstrations can be put into a single prompt to reduce complexity. It can reduce the number of prompts and LLM calls to 1. In consequence, it need not collect and manage the outputs of multiple calls of LLMs, because the output of this prompt is a sequence of discriminations. Second, a single prompt means a single inference time in theory. Also, it only needs 1 instruction and saves $M_d - 1$ instruction tokens compared with the basic version. The token number saved is proportional to the dataset size. The larger the dataset size is, the more tokens are saved in this way. This form is called **Single Prompt Multiple Discriminations Style** (S-M Style), which increases the scalability of JD-SCOPE. Last, together with more demonstrations, the multiple current questions of the candidate column pairs can provide a richer context in the prompt. LLMs can see more comprehensive information in the *S-M style* prompt and output consistent results for the current questions, exhibiting insensitivity to thresholds. Therefore, the imitated version prompt in Fig. 2 (III) evolves into the S-M style from the basic version.

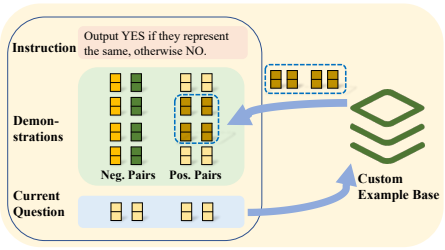

Figure 6: An illustrations of prompt building process with *Custom Example Base* for semantic join discovery.

2.5 Semantic Join Discovery Adaption JD-SCOPE can also be used on semantic join discovery. Column pairs that are semantic joinable are correlated by deterministic many-to-many mapping relations (e.g., *country code* and *language code*). It is tricky to find these pairs in data lakes. However, LLMs are few-shot learners [34]. As shown in Fig. 6, to conduct the semantic join discovery, a few semantic join discovery examples should be first collected as a *Custom Example Base*, which is another source of demonstrations in prompts to guide the LLMs to understand the semantic join discovery. These examples are embedded [1] as vectors for the subsequent search. Second, construct a normal prompt following the standard JD-SCOPE pipeline. Third, the current question of the prompt is also embedded in the same way. Then, query examples (e) in the *Custom Example Base* that are similar to the current question (q) according to the pair similarity $S(q, e)$ based on the em-

[1] Only the column names are embedded for simplicity. The embedding model used is Table Embedding 150 dim [21]

Table 1: Sample Distribution of **JD-Lake**.

Partition	#Positive	#Negative	Total
Normal Set	1384	1303	2687
Semantic Set	764	764	1528
Blend Set	1303	1303	2606

beddings to choose suitable demonstrations for prompts.

(2.1)
$$S(q,e) = \max\{s(q_1,e_1), s(q_1,e_2), s(q_2,e_1), s(q_2,e_2)\},$$

where q_i and e_j $(i \in \{1,2\}, j \in \{1,2\})$ are embeddings of one item in the pair, $s(q_i, d_j)$ is the cosine similarity. Examples similar to the current question can better support semantic join discovery compared with randomly selected ones. Finally, replace the examples in the demonstration block of the normal prompt with the searched samples to obtain the final prompt.

3 Experiments

This section will first introduce the datasets used and experimental settings, and then verify ***how*** effective JD-SCOPE is, further illustrate ***which*** module of it overcome what kind of challenges, and explain ***why*** they are effective through empirical experiments.

3.1 Datasets Description

3.1.1 Deficiencies in Existing Datasets The existing join discovery datasets have flaws that render the evaluation unfair. **Nextia$_{\mathbf{JD}}$** [19, 20] is a dataset that supports data discovery over data lakes. However, it uses pseudo labels based on the containment similarity (column overlap) and the length ratio of two columns (The training set contains a few labeled samples). This way of labeling is biased, making the evaluation results more in favor of overlap-based baselines. The **Sigma** [10] dataset was used in [10], but it is not accessible to the public and is only open to Sigma Computing accounts. The **Spider** [44] is a text-to-SQL dataset, and can also serve as a join discovery evaluation dataset after parsing the schema SQL file and retrieving the paths between primary keys and foreign keys as ground truth [10]. However, there are only 772 joinable pairs in this dataset, which is too small. Evaluation of this dataset is prone to large variance and is not convincing. **Table Union Search Benchmark** [31] is used to evaluate union search and can also be used as a benchmark for join discovery. However, this dataset is synthesized by performing selection and projections on Open Data [1]. Different tables may share the schema information, which destroys the naturalness of the data.

3.1.2 The Proposed Benchmark To unbiasedly demonstrate the effectiveness of different methods, a real-word dataset annotated manually is proposed to evaluate the **J**oin **D**iscovery on the data **Lake**, named **JD-Lake**. **JD-Lake** is inspired by Nextia$_{\text{JD}}$ [19, 20] to obtain data from copy-free open data repositories such as Kaggle and OpenML. JD-Lake has three partitions, i.e., *Normal Set*, *Semantic Set* and *Blend Set*. Seven participants (3 females, 4 males) annotated the dataset in multiple rounds via crowdsourcing. The number of positive and negative samples of each partition is well balanced as present in Tab. 1. JD-SCOPE and baseline methods will be compared on these three datasets. The *Normal Set* is used to evaluate the join discovery while the *Semantic Set* is for the semantic join discovery. The *Blend Set* is designed to validate overall capability.

3.2 Experimental Settings

3.2.1 Baselines Comparison approaches include multiple classic overlap-based methods, i.e., Jaccard-Similarity-Join (**JS**), Jaccard-Containment-Join (**JC**), Intersection-Join (**IS**), Edit-Distance-Jaccard-Containment-Join (**ED-JC**) and Edit-Distance-Intersection-Join (**ED-JS**). The latest **WarpGate** [10] and its variants which switch the embedding model from Table Embedding [21] to BERT [12] or GPT [7]) are also compared. The similarity from above methods will be compared with a finetuned threshold. If similarity surpasses threshold, columns are categorized as joinable otherwise dis-joinable.

JD-SCOPE is evaluated with a series of the most popular LLMs, spanning the PaLM2 [2] from Google, the Llama2 [37] from Meta, the Claude [3, 4] from Anthropic, the Text-Davinci-003, the ChatGPT and the GPT-4 from OpenAI.

3.2.2 Search and Evaluations Join discovery is formalized as a binary classification as in [19, 20]. The accuracy (Acc.), precision (Pre.), recall (Rec.) and F_1-score (F_1) are used to measure the performance. The threshold of each method is tuned for the highest F_1 score. The number of demonstrations is fixed to 2 aiming to balance performance and context window size. The number of sampled values representing each column is fixed to 3 across this paper. The temperature is 0.9 for diversity during search and evaluation, and 0 for reproducibility during testing. A total of 54 rounds are conducted, averaging 9 rounds per task form.

3.3 Methods Comparison All methods are first evaluated on the *Normal Set*, and the results are shown in Tab. 2. As can be seen from the first block, IS-Join achieves the highest F_1 score of 76.90 among all

Table 2: The performance comparison on *Normal Set*.

Method	Acc.	Pre.	Rec.	F_1
JS-Join	69.96	96.60	43.17	59.67
JC-Join	71.18	86.64	52.06	65.04
IS-Join	79.37	90.84	66.67	76.90
ED-JC-Join	71.85	78.22	62.83	69.69
ED-IS-Join	75.13	76.38	74.84	75.60
WarpGate (C2C)	83.43	89.61	76.72	82.66
WarpGate (C2H)	55.40	53.59	99.78	69.73
WarpGate (H2H)	85.63	95.11	75.99	84.49
Bert (C2C)	53.31	52.45	99.86	68.77
Bert (C2H)	51.82	51.66	100.0	68.13
Bert (H2H)	58.53	55.46	98.84	71.05
GPT (C2C)	71.71	66.88	89.23	76.46
GPT (C2H)	56.37	54.31	96.02	69.38
GPT (H2H)	80.19	78.61	84.53	81.46
APE† (Text-Davinci-003)	71.57	71.68	74.06	72.85
JD-SCOPE (Llama2-13B)	55.12	56.43	56.43	56.43
JD-SCOPE (Llama2-13B-Chat)	47.19	43.94	9.18	15.18
JD-SCOPE (Claude-V1)	48.65	0	0	0
JD-SCOPE (Claude-V2)	56.35	75.18	22.76	34.94
JD-SCOPE (PaLM2-Text)	71.19	92.48	47.98	63.18
JD-SCOPE (PaLM2-Chat)	46.19	45.02	20.23	27.92
JD-SCOPE (ChatGPT)	72.68	71.75	77.46	74.50
JD-SCOPE (GPT-4)	90.03	92.15	88.15	90.10
JD-SCOPE (Text-Davinci-003)	**90.66**	87.29	95.81	**91.35**

overlap-based methods. Ranked second is ED-IS-Join, indicating that fuzzy join by edit distance did not bring better results when combined with IS-Join. However, when combined with JC-Join, the fuzzy join works and ED-JC-Join exceeds JC-Join 4.65 on F_1 score. JC-Join is better than JS-Join, confirming the conclusion in [48].

The second block contains the embedding-based methods. WarpGate (H2H) achieves the best results of F_1 score 84.49, showing the effectiveness of metadata, which **further supports** the rationale for the existence of *Metadata Generation*. The third block contains the results of JD-SCOPE with different LLMs. JD-SCOPE (Text-Davinci-003) reaches the best performance of all baselines, 91.35 on F_1 score and 90.66 on accuracy, a 6.86 improvement over the previous SoTA WarpGate (H2H). It also exceeds the APE† 18.5 on F_1 score, demonstrating the advantage of joint search space over that only contains the instruction dimension. Although GPT-4 is a larger model, it does not achieve higher results. It can be seen that the non-chat version LLMs **consistently outperform** their chat version counterparts (i.e., PaLM2-Text > PaLM2-Chat, Text-Davince-003 > ChatGPT, etc).

The best of the above three types of methods are JD-SCOPE (Text-Davinci-003), WarpGate (H2H) and IS-Join. They are further analyzed under different thresholds (Thresholds for IS-Join were scaled to 1 for comparison purposes.) and shown in Fig. 7 (I). JD-SCOPE (Text-Davinci-003) is higher than other

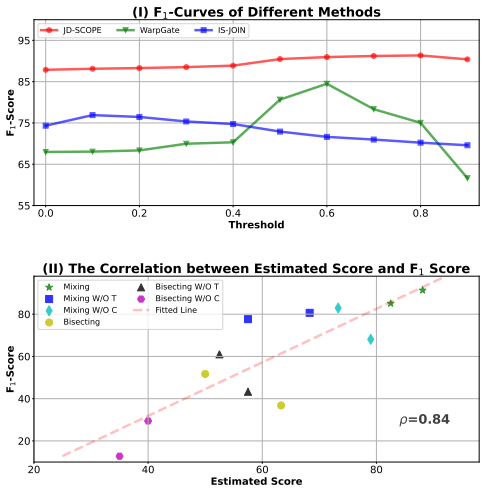

Figure 7: Performance comparison and analysis. (I) F_1 curves of three best methods under different thresholds. (II) Correlations of estimated scores and F_1 scores. The pink dashed line is the fitted line of these candidate dots.

Table 3: The module effectiveness study of JD-SCOPE on *Normal Set* with Text-Davinci-003.

Method	Acc.	Pre.	Rec.	F_1
JD-SCOPE	**90.66**	87.29	95.81	**91.35**
- Candidate Column Selection	83.44	87.83	78.76	83.05
- Trivial Pair Elimination	89.84	85.50	96.68	90.74
- Meta Generation	67.58	66.69	74.06	70.18
- Auto Template Exploration	71.57	71.68	74.06	20.15
- Output Result Stabilization	86.16	97.38	75.14	84.83

baselines at all thresholds, which further demonstrates its superiority. The curve of JD-SCOPE (Text-Davinci-003) is more stable than that of WarpGate. This reflects JD-SCOPE is insensitive to thresholds, which is very **practical** in industrial applications.

3.4 The Role of Each Module

To open the black box of JD-SCOPE and figure out why it works logically, some ablation studies are conducted to see the principles. The SoTA is achieved with the *Mixing Form*, so the following experiments use this task form by default, except for analyses specific to task forms. The experimental results are gathered in Tab. 3.

3.4.1 Candidate Column Preparation The related factors are collected in the second block. The "-*Candidate Column Selection*" is conducted by removing the demonstrations in the *Search and Evaluation* and testing phases, compared to SoTA JD-SCOPE. F_1 score drops to 83.05 in the absence of unsupervised demonstrations because in-context learning ability can not be utilized. The *Trivial Pair Elimination* is then removed in testing phases and the F_1 score also drops to 90.74.

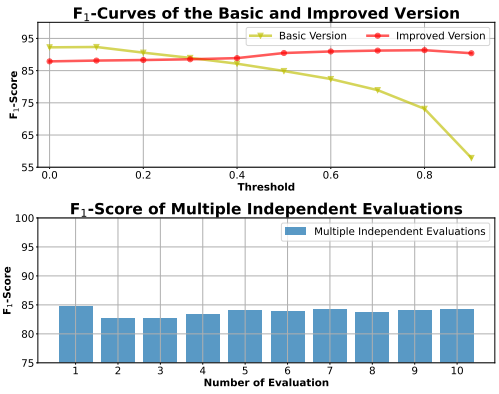

Figure 8: Illustrations of F_1 curves of basic and improved multiple discriminations and the F_1 scores bar graph of multiple independent evaluations ($M_d = 1$).

Eliminating the trivial pairs can offer more informative examples and improve the quality of *Demonstrations Base* and then improve join discovery. The *Metadata Generation* module is also removed and the F_1 score was lowered to 70.18, proving the key role it plays in join discovery task as discussed in Sec. 2.2.2. These sub-modules take advantage of tabular data to create effective pseudo labels, overcoming the challenge of insufficient labeled samples on data lakes.

3.4.2 Auto Template Exploration The optimal template from *Auto Template Exploration* is replaced with that in Fig. 2 (III). The F_1 score is down sharply to 20.15 in the third block, proving it is the most crucial factor. As discussed in Sec. 2.3.1 and 2.3.2, the optimal instruction can help LLMs understand the join discovery task and the suitable task form can enhance the thinking logic of LLMs. The template in Fig. 2 (III) is casual and *Auto Template Exploration* can overcome the challenge of finding suitable prompt templates.

To validate the ranking effectiveness of this module, some candidates are selected at equal intervals based on estimated scores and to cover diverse task forms. These candidates are evaluated on *Normal Set* to obtain corresponding F_1 scores and they are plotted in Fig. 7 (II). It is evident that there is a positive correlation between the estimated score and the F_1 score in terms of **trend**, and Pearson's correlation coefficient is as high as 0.84. It empirically proves the effectiveness of the *Validation Set* although it is constructed in an unsupervised way.

3.4.3 Output Result Stabilization This strategy is removed from the SoTA JD-SCOPE in *Search and Evaluation* and testing phases, and the F_1 score dramatically falls to 84.83. In addition, the performance of multiple independent evaluations of JD-SCOPE with $M_d = 1$ It shows from the bar graph that F_1 scores of 10 independent evaluations are all under 85, not high results. As seen in the line chart, the basic version of multiple discriminations (yellow curve) can achieve higher results at the proper threshold. However, it is sensitive to the threshold, which may vary with data.

Table 4: The performance comparison on *Semantic Set*.

Method	Acc.	Pre.	Rec.	F_1
JS-Join	49.08	6.25	0.13	0.26
JC-Join	46.07	10.53	1.05	1.90
IS-Join	44.76	33.87	10.99	16.60
ED-JC-Join	48.63	49.28	94.11	64.69
ED-IS-Join	65.18	61.24	82.72	70.38
WarpGate (C2C)	57.98	54.52	96.34	69.63
WarpGate (C2H)	65.05	61.98	77.88	69.03
WarpGate (H2H)	61.45	58.09	82.20	68.08
Bert (C2C)	50.00	50.00	100.0	66.67
Bert (C2H)	50.00	50.00	100.0	66.67
Bert (H2H)	50.00	50.00	100.0	66.67
GPT (C2C)	50.00	50.00	100.0	66.67
GPT (C2H)	50.00	50.00	100.0	66.67
GPT (H2H)	50.00	50.00	100.0	66.67
JD-SCOPE (Text-Davinci-003)	**83.44**	84.57	81.81	**83.17**
- Custom Example Base	64.53	68.62	53.53	60.15

3.5 Results on Semantic Join Discovery Semantic join discovery ability is evaluated on the *Semantic Set* and shown in Tab. 4. Clearly, all the equal matching methods in the first block fail on this task, which achieves very low F_1 scores. The fuzzy match based methods significantly improve the F_1 score, but they remain low. Because equal matching or fuzzy matching in syntactic can not reveal the conceptually correlated relationships. The embedding-based methods in the second block, WarpGate can only yield 68-70 on F_1 score, while Bert and GPT almost totally fail on this task. JD-SCOPE outperforms the second place method by a large margin, achieving 83.17 on F_1 score, with the help of *Custom Example Base*. If the base is removed, the F_1 score will jump to 60.15, which shows the importance of the *Custom Example Base*. JD-SCOPE further unleashes the in-context learning capability of LLMs by the *Custom Example Base*, overcoming the expansion issue on semantic join discovery.

4 Conclusion

This paper attempts to apply LLMs to explore data relationships in data lakes which handles complex cases with a basic filter for simple ones in real-world applications. It identifies the migration problem and corresponding four challenges of this practice and then proposes the novel JD-SCOPE to overcome them respectively. JD-SCOPE outlines a futuristic join discovery architecture with LLMs, shaping a new tool through domain integration, showing significant potential impact.

References

[1] Data.gov, 2023. URL https://data.gov/. Accessed: 2023-10-05.

[2] Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. Palm 2 technical report, 2023.

[3] anthropic. Introducing claude, 2023. URL https://www.anthropic.com/index/introducing-claude. The official website of claude.

[4] anthropic. Claude2, 2023. URL https://www.anthropic.com/index/claude-2. The official website of claude2.

[5] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

[6] Irénée-Jules Bienaymé. *Considérations à l'appui de la découverte de Laplace sur la loi de probabilité dans la méthode des moindres carrés*. Imprimerie de Mallet-Bachelier, 1853.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[8] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

[9] Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, pages 493–507, 1952.

[10] Tianji Cong, James Gale, Jason Frantz, H. V. Jagadish, and Çagatay Demiralp. Warpgate: A semantic join discovery system for cloud data warehouses. In *13th Conference on Innovative Data Systems Research, CIDR 2023, Amsterdam, The Netherlands, January 8-11, 2023*. www.cidrdb.org, 2023. URL https://www.cidrdb.org/cidr2023/papers/p75-cong.pdf.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019. URL https://api.semanticscholar.org/CorpusID:52967399.

[13] Haoyu Dong and Zhiruo Wang. Large language models for tabular data: Progresses and future directions. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2997–3000, 2024.

[14] Yuyang Dong, Kunihiro Takeoka, Chuan Xiao, and Masafumi Oyamada. Efficient joinable table discovery in data lakes: A high-dimensional similarity-based approach. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 456–467. IEEE, 2021.

[15] Yuyang Dong, Chuan Xiao, Takuma Nozawa, Masafumi Enomoto, and Masafumi Oyamada. Deepjoin: Joinable table discovery with pre-trained language models. *arXiv preprint arXiv:2212.07588*, 2022.

[16] Bradley Efron. *Bootstrap methods: another look at the jackknife*. Springer, 1992.

[17] Huang Fang. Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem. In *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 820–824. IEEE, 2015.

[18] Xi Fang, Weijie Xu, Fiona Anting Tan, Jiani Zhang, Ziqing Hu, Yanjun Jane Qi, Scott Nickleach, Diego Socolinsky, Srinivasan Sengamedu, Christos Faloutsos, et al. Large language models (llms) on tabular data: Prediction, generation, and understanding-a survey. 2024.

[19] Javier Flores, Sergi Nadal, and Oscar Romero. Scalable data discovery using profiles. *ArXiv*, abs/2012.00890, 2020. URL `https://api.semanticscholar.org/CorpusID:227247800`.

[20] Javier de Jesús Flores Herrera, Sergi Nadal Francesch, and Óscar Romero Moral. Towards scalable data discovery. In *Advances in Database Technology: EDBT 2021, 24th International Conference on Extending Database Technology: Nicosia, Cyprus, March 23-26, 2021: proceedings*, pages 433–438. OpenProceedings, 2021.

[21] Michael Günther, Maik Thiele, Julius Gonsior, and Wolfgang Lehner. Pre-trained web table embeddings for table discovery. In *Fourth Workshop in Exploiting AI Techniques for Data Management*, pages 24–31, 2021.

[22] Rihan Hai, Sandra Geisler, and Christoph Quix. Constance: An intelligent data lake system. In *Proceedings of the 2016 international conference on management of data*, pages 2097–2100, 2016.

[23] Yeye He, Kris Ganjam, and Xu Chu. Sema-join: joining semantically-related tables using big table corpora. *Proceedings of the VLDB Endowment*, 8 (12):1358–1369, 2015.

[24] Wassily Hoeffding. On the distribution of the number of successes in independent trials. *The Annals of Mathematical Statistics*, pages 713–721, 1956.

[25] Bill Inmon. *Data Lake Architecture: Designing the Data Lake and avoiding the garbage dump*. Technics publications, 2016.

[26] Pwint Phyu Khine and Zhao Shun Wang. Data lake: a new ideology in big data era. In *ITM web of conferences*, volume 17, page 03025. EDP Sciences, 2018.

[27] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

[28] Peng Li, Xiang Cheng, Xu Chu, Yeye He, and Surajit Chaudhuri. Auto-fuzzyjoin: auto-program fuzzy similarity joins without labeled examples. In *Proceedings of the 2021 International Conference on Management of Data*, pages 1064–1076, 2021.

[29] Natalia Miloslavskaya and Alexander Tolstoy. Big data, fast data and data lake concepts. *Procedia Computer Science*, 88:300–305, 2016.

[30] Fatemeh Nargesian, Ken Q Pu, Erkang Zhu, Bahar Ghadiri Bashardoost, and Renée J Miller. Optimizing organizations for navigating data lakes. *arXiv preprint arXiv:1812.07024*, 2018.

[31] Fatemeh Nargesian, Erkang Zhu, Ken Q Pu, and Renée J Miller. Table union search on open data. *Proceedings of the VLDB Endowment*, 11(7):813–825, 2018.

[32] Fatemeh Nargesian, Erkang Zhu, Renée J Miller, Ken Q Pu, and Patricia C Arocena. Data lake management: challenges and opportunities. *Proceedings of the VLDB Endowment*, 12(12):1986–1989, 2019.

[33] Paul Ouellette, Aidan Sciortino, Fatemeh Nargesian, Bahar Ghadiri Bashardoost, Erkang Zhu, Ken Q Pu, and Renée J Miller. Ronin: data lake exploration. *Proceedings of the VLDB Endowment*, 14(12), 2021.

[34] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[35] Jie Song and Yeye He. Auto-validate: Unsupervised data validation using data-domain patterns inferred from data lakes. In *Proceedings of the 2021 International Conference on Management of Data*, pages 1678–1691, 2021.

[36] Zhoujin Tian, Chaozhuo Li, Zhiqiang Zuo, Zengxuan Wen, Xinyue Hu, Xiao Han, Haizhen Huang, Senzhang Wang, Weiwei Deng, Xing Xie, et al. Multi-grained topological pre-training of language models in sponsored search. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2189–2193, 2023.

[37] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[38] Quan M Tran, Suong N Hoang, Lam M Nguyen, Dzung Phan, and Hoang Thanh Lam. Tabularfm: An open framework for tabular foundational models. *arXiv preprint arXiv:2406.09837*, 2024.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[40] Jiannan Wang, Guoliang Li, and Jianhua Feng. Extending string similarity join to tolerant fuzzy token matching. *ACM Transactions on Database Systems (TODS)*, 39(1):1–45, 2014.

[41] Yaohua Wang, FangYi Zhang, Ming Lin, Senzhang Wang, Xiuyu Sun, and Rong Jin. Robust graph structure learning over images via multiple statistical tests. *arXiv preprint arXiv:2210.03956*, 2022.

[42] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[43] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[44] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*, 2018.

[45] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[46] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

[47] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022.

[48] Erkang Zhu, Fatemeh Nargesian, Ken Q Pu, and Renée J Miller. Lsh ensemble: Internet-scale domain search. *arXiv preprint arXiv:1603.07410*, 2016.

[49] Erkang Zhu, Dong Deng, Fatemeh Nargesian, and Renée J Miller. Josie: Overlap set similarity search for finding joinable tables in data lakes. In *Proceedings of the 2019 International Conference on Management of Data*, pages 847–864, 2019.