Predicting Table Joinability in Data Lakes using a Metadata Knowledge Graph

Sola Shirai
IBM
Yorktown Heights, NY, USA
solashirai@ibm.com

Oktie Hassanzadeh
IBM
Yorktown Heights, NY, USA
hassanzadeh@us.ibm.com

Debarun Bhattacharjya
IBM
Yorktown Heights, NY, USA
debarunb@us.ibm.com

Gaetano Rossiello
IBM
Yorktown Heights, NY, USA
Gaetano.Rossiello@ibm.com

ABSTRACT

Discovering joinable tables in data lakes is a crucial task for systems that provide insights to users such as business analysts, data engineers, and data scientists. Due to the lack of unifying schema for such data lakes, it is typically necessary to require some level of access to the contents of tables to inspect the contents of table columns and discover joinable pairs. We introduce an alternate approach for tackling this problem, wherein we utilize only the textual metadata of tables to predict joinability. Our approach, MeGNN-Join (**Me**tadata **G**raph **N**eural **N**etwork for **Join**ability Prediction), models table metadata as a knowledge graph and frames predicting table joinability as a link prediction task over the graph. MeGNN-Join's lack of reliance on table contents allows it to perform predictions in situations where typical content-based methods cannot operate, and it utilizes semantic enrichment of missing metadata to produce more effective predictions. To validate our approach, we design an evaluation framework using real world data from open government data portals and demonstrate that MeGNN-Join shows superior performance and generalizability to new tables compared to baselines.

VLDB Workshop Reference Format:
Sola Shirai, Oktie Hassanzadeh, Debarun Bhattacharjya, and Gaetano Rossiello. Predicting Table Joinability in Data Lakes using a Metadata Knowledge Graph. VLDB 2026 Workshop: Tabular Data Analysis (TaDA).

VLDB Workshop Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://zenodo.org/records/20143332.

1 INTRODUCTION

As data has become increasingly available, so too have we seen an increased need to effectively manage and utilize this data in various data science and analytics pipelines. For example, a data lake may contain large collections of data with similar concepts, but it may not be easy to identify relationships between them due to the lack of a uniform underlying schema.

Within this problem space, we are particularly interested in how *tabular data* within data lakes can be utilized more effectively. The well-structured nature of tabular data makes it relatively easy to expand on and aggregate data, for instance, by identifying columns from two tables which refer to the same concept in order to join the tables. Performing such a join operation is an important tool to fully utilize data that has been curated from different sources and provide a more comprehensive analysis. However, the lack of uniform schema across data lake tables makes it difficult to identify joinable tables. Columns with the same name in different tables may refer to different concepts – e.g., a "name" column in a table about county-level demographics versus a company's employee table. A similar challenge exists when two tables refer to the same concept using different labels – e.g., "Community Board" and "CD Number" refer to the same concept (the number assigned to the community district), but this would not be immediately obvious based on only the column names. The typical way to approach these challenges is to inspect the contents of tables rather than just the column names, as finding matching entries under a given column in two tables serves as evidence that those columns refer to the same concept. Work in this area typically explores topics such as efficient identification of joinable columns [7, 8, 12, 21, 44] and joinability on different types of data [28, 32].

The need to process table contents in these typical methods, however, may be undesirable in many real world applications. Some governance structures and access guidelines for data lakes may prohibit access to table contents – this makes traditional approaches for joinable table discovery infeasible. Furthermore, processing and storage of table contents to perform joinable table discovery may be expensive, especially as the scale of data lakes increases. While existing work has explored how to address factors such as computational cost or privacy concerns, in this work we are interested in considering an altogether different approach which could entirely avoid potential issues related to table contents. In order to predict table joinability without the use of table contents, we turn our sights to the *metadata* that is attached to tabular data.

In this paper, we propose MeGNN-Join (**Me**tadata **G**raph **N**eural **N**etwork for **Join**ability Prediction) – our approach for predicting table joinability using *only* the tables' metadata. An overview of our method is shown in Figure 1. Metadata serves as a means to

Proceedings of the VLDB Endowment. ISSN 2150-8097.

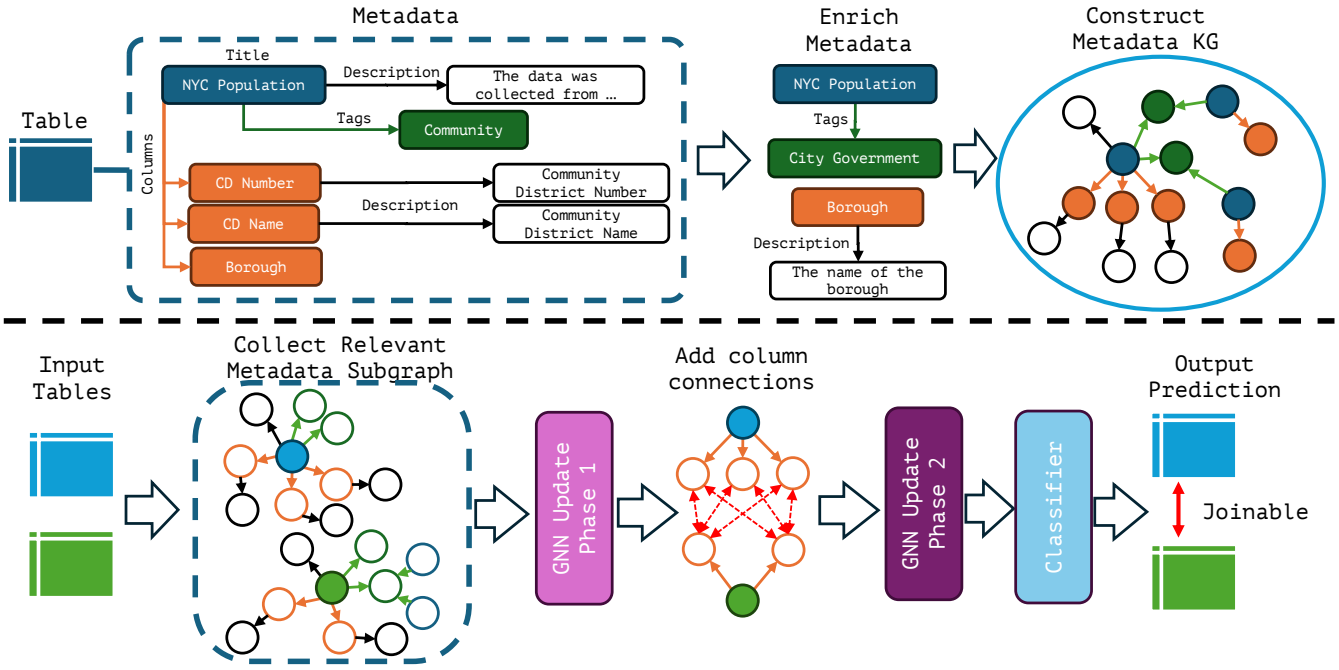

Figure 1: A high-level overview of MeGNN-Join, depicting how table metadata is enriched and converted into a KG (top) and how we process metadata graphs using GNNs to enable joinability prediction (bottom).

more easily get a high-level understanding of the data without the need to inspect its contents, and is often made easily available from enterprise data catalogs[1] or open data portals. Even if accessing and/or analyzing the contents of tables is costly or disallowed, such metadata may be utilizable to get a rough prediction of whether two tables are joinable.

We frame the problem of joinability prediction as a link prediction task, through (1) modeling tables and their metadata as a knowledge graph (KG), (2) utilizing semantic enrichment to generate any missing pieces of metadata, (3) using language models to produce features from textual metadata, and (4) processing the metadata KG using a multi-phase approach. We construct our metadata KG by connecting table names, column names, tags/categories, and descriptions for each table, and train two phases of graph neural network (GNN) [29] modules to produce table representations, which can be used to predict joinability between a pair of tables.

Our contributions in this paper are as follows: (1) we propose a novel approach for predicting table joinability, which frames the problem as a link prediction task over a KG of table metadata; (2) we present MeGNN-Join, which effectively utilizes and processes a KG of textual metadata to predict table joinability; (3) we devise an evaluation framework using tabular data from three open government data portals, as well as one synthetic dataset; (4) we evaluate MeGNN-Join against a variety of baselines, encompassing other text-embedding methods and common link prediction approaches over three evaluation settings, demonstrating the effectiveness of MeGNN-Join, especially for more difficult joins and for tables which were unseen during training.

[1]Examples of data governance catalogues for data lakes can be seen in services such as AWS, Databricks, and Dremio.

2 RELATED WORK

A number of recent surveys and works exist on the problem of *data discovery* [5, 11, 26], which becomes an important problem as the number, format, and provenance of tables present in data lakes increases. The lack of unifying, overarching schema that defines and relates concepts across various tables is a central problem. While past work has explored the topic of connecting columns to KG concepts for data management (which in turn could be applied to identify joinable tables) they either require manual effort [6, 10] or rely on table contents to aid in automatic linking [16].

Several recent methods have explored the task of table joinability, often with a strong focus on improving efficiency in memory usage or set overlap detection. Some examples include JOSIE [44], which introduces an algorithm for efficient set overlap discovery, PEXESO [7], which uses column embeddings to determine joinability using high-dimensional similarity metrics, and work that explores the use of prefix trees [21] to effectively prune multi-attribute data. There has also been exploration of table joins specifically on numeric data [28, 32], joinable table discovery using pre-trained language models [8], and utilizing large-scale datasets to identify semantic similarity between tables [14, 20]. Bharadwaj et al. [2] leveraged metadata from large-scale query logs to discover related columns across data streams – in contrast, we look only at tables in isolation without such usage information. The biggest difference of our method compared to such previous works is that all of these works assume the

use of table contents, while we use *only* the metadata about the tables to make predictions.

There also has been some exploration of the use of graphs and metadata for various downstream tasks. Recently, Sequeda et al. show the effectiveness of using a KG in question answering over a relational database through the use of a manual mapping of the database to a KG [30]. ATJ-Net [1] constructs hypergraphs of tables and applies a GNN to perform prediction tasks such as classification or recommendation – however, this method relies on well-defined primary- and foreign-key relations for tables. KGLac [15] demonstrated a method which involved constructing a knowledge graph to capture semantics and metadata about tables. CARTE [17] explored an approach to utilize KGs as a source of pre-training for table representation learning. Past work has explored the use of string similarity to perform table joins as well, as seen in surveys such as [37, 41]. However, the primary focus of such works has been the string similarity of table cells rather than table metadata.

3 PRELIMINARIES

3.1 Table Joinability

The main problem of our work is to predict if two tables are joinable. In a set of tables following a well-defined schema, joinability can be determined by foreign key relationships pre-defined in the schema. Within a data lake, and in the absence of a comprehensive *schema discovery* (or *schema extraction* [31]) process – which could be highly challenging and prohibitively expensive – we must identify potentially joinable tables using other available information.

In this paper, we follow related work [7] and define two tables to be joinable if they have a pair of columns whose similarity is above some threshold, according to a content-based similarity measure. Similarity then is computed using set-based measures like set overlap or containment ratio – columns being similar implies that they contain at least some cell values on which their contents can be joined.

3.2 Metadata

We consider any textual data about a table besides its cell values to be metadata. In this work, we consider the following pieces of metadata to be relevant to our task: the **table name and description**; the **column names and descriptions** for each column in the table; the **tags**, or set of strings indicating some categorization to which the table is relevant; and **row labels**, which are text descriptions of the concept which each row in the table represents. Of these pieces of metadata, we assume to always have table and column names available. Besides these, the availability of metadata can vary on a table-to-table basis and based on the source from which tables are collected. Row labels are rarely available, or only available indirectly through table descriptions.

3.3 Semantic Enrichment

While semantic information and metadata of tabular data can be quite useful, it is also often incomplete or missing in real-world data. This, in turn, can be a limiting factor surrounding methods which aim to heavily make use of metadata like our work. To address this issue we explore recent advances in the use of large language models (LLMs) for semantic enrichment of metadata [22, 25] to automatically generate and enrich table metadata.

In this work, we focus on three semantic enrichment tasks. The first is **column name expansion**, where cryptic or abbreviated column names are expanded into more human readable forms. The second is **description generation**, where missing natural languages descriptions are generated for a given column. Lastly, we utilize **tag generation** to generate keywords which can act as tags or high-level categorizations of tables.

3.4 Metadata Knowledge Graph

To explore our approach of predicting table joinability using only metadata, we chose to encode the metadata of tables in a knowledge graph. Using a graph structure allows us to be flexible with the availability of data, such as tables missing or having large amounts of a particular type of metadata, rather than requiring some fixed feature size. We also wish to leverage information about *tags*, which we consider to be reliable but incomplete indications about the concepts contained in tables. We define our knowledge graph $G = (E, R, T)$ as a set of entities E, relations R, and triples T. The triples in T are of the form (h, r, t) where $h, t \in E$ and $r \in R$. As we are working with textual metadata, each entity $e \in E$ has a corresponding string of text, which will serve as its initial input feature which we will utilize in our approach.

4 MEGNN-JOIN

Our method for predicting table joinability frames the problem as a link prediction task in a KG of table metadata. Given two target tables, we produce a joinability score and repeat the process for multiple pairs of tables to produce a ranking. An overview of our approach is shown in Figure 1.

4.1 Metadata Enrichment

The first step of our method is to perform semantic enrichment to fill in missing metadata and generate additional keywords to support our prediction model. Each enrichment task is highly contextual – e.g., decyphering cryptic columns depends on information about what the other columns in the table are, as well as background context like the table's name and description. Following in the steps of related work [42] which supports the utility of LLMs to effectively expand abbreviations or acronyms in column names, we make use of an LLM to perform our semantic enrichment steps.

For a given table t we first collect all of its available metadata. This includes the table's name, description, a set of keywords or tags, its column names, and column descriptions. Often, some of this metadata is missing or incomplete – the column names may be cryptic and hard to understand by humans or AI systems, column descriptions may be missing entirely, and tags may be incomplete or lacking in granularity. To generate missing metadata, the available metadata is collected and formatted into a prompt to subsequently input to an LLM model (full prompt details are available in Section 1 of our Supplemental Material). In our experiments, we use a Llama 3.3 70b [9] model to generate structured textual output, corresponding to our three semantic enrichment tasks.

An example of our semantic enrichment task is illustrated in Figure 2. In this table, curated from the Canada.ca open data portal,

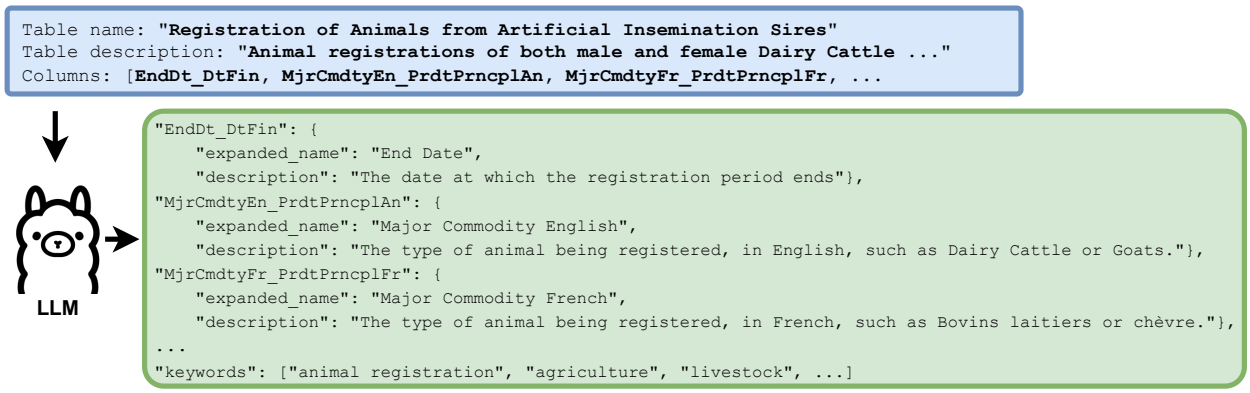

Figure 2: An example of inputs and outputs for our semantic enrichment tasks. Long text and outputs are omitted, indicated with ellipses.

the original metadata is missing column descriptions and contains multiple abbreviations in its original column names. These column names are difficult to interpret at first glance – due to the table containing information in both English and French, the column names are also presented using abbreviations in both languages (e.g., `MjrCmdtyEn_PrdtPrncplAn` is composed of "Major Commodity English" and "Produit Principal Anglais"). We can see how the LLM is able to successfully expand the cryptic names and also provide descriptions which help to better understand the content of the columns. Further, the expansion simplifies the column name to only include the English portion.

The output of this enrichment step is a set of expanded column names, new column descriptions, and keywords associated with the table. We store and make use of the expanded column names in place of the original column names for all subsequent steps. For column descriptions, we only utilize the generated descriptions if none were available in the original data, given that descriptions often indicate some level of human effort involved in curating the metadata and believe it to be a more reliable source of information than generated content. For generated keywords, we add and use all newly generated keywords together with any tags which were present in the original metadata.

This semantic enrichment step can be completed once off-line, and the resulting enrichments are stored for future use.

4.2 Metadata KG Construction

Given the textual metadata for a table, we construct a KG representation for it by initializing entities for each piece of metadata. If all types of metadata are available then we would produce entities for the table name, table description, each column name, each column description, each tag, and the row label. We consider the table name as the main entity for the table within our KG. Additionally, we assume that all tags in the metadata to refer to the same concept within a given data portal. On the other hand, each column and table is treated as a distinct entity, regardless of whether other columns or tables have the same name. This is to enforce our assumption that two columns in different tables with the same name do not necessarily refer to the same concept, and it allows us to uniquely attach column descriptions to columns in different tables. Through this process we construct a KG with 4 properties, connecting the table entities to each of its metadata entities and connecting columns to their descriptions. We also add a reflexive property to capture joinability links between tables, to represent any known table joins within the dataset.

For each entity in this metadata KG, we convert its text feature into an embedding using a pre-trained text embedding model. The model takes as input one or more sentences, corresponding to an entity in the metadata KG, and output an embedding representing those sentences. The model needs to be capable of handling longer texts in the metadata, such as descriptions and table names. In our implementation, we choose a Sentence-BERT [27] (SBERT) model. For each entity $e \in E$, this process produces corresponding a vector $\vec{e} \in \mathbb{R}^F$, where F is the dimension of the text embedding.

Through this process we construct a KG representation of the metadata for tables, which we further process through subsequent GNN modules. Metadata that is seen during training can be stored at this point for future use, and the KG representation for new table metadata can be easily computed online.

4.3 MeGNN-Join Model

MeGNN-Join operates on a pair of tables to produce predictions – in the remainder of this section we will refer to the pair of tables as the *query* table, t_Q, and *candidate* table, t_C, where the query table is a specific table for which we want to produce joinability scores and the candidate table is one of a set of tables for which we want to determine joinability with the query table. Our method details are explained for a single query and candidate table pair, and the process is repeated for multiple candidate tables.

For the pair of tables t_Q and t_C, we collect their 2-hop neighborhood in the metadata KG to serve as the relevant subgraph to feed into subsequent steps. This neighborhood encompasses all of the metadata for each table, as well as some links between tags and other tables that share that tag. We denote these subgraphs as G_Q and G_C, and the set of columns for each table as C_Q and C_C.

The core idea for the MeGNN-Join model now is to produce embeddings for tables such that tables that are likely to be joinable

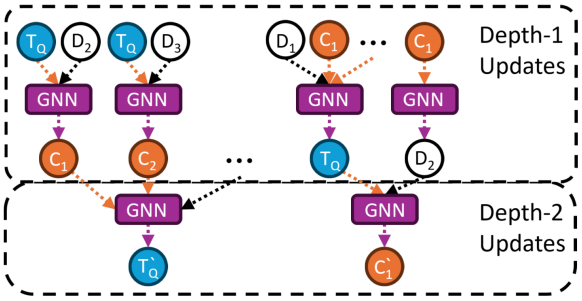

Figure 3: Example of embedding updates using GNNs. D_i, c_i, **and** T_i **indicate description, column, and table nodes.**

are more similar than tables that are not joinable. To do so, our model implements two phases of updates for embeddings in the KG. In the first phase, the embeddings of tables and columns are updated based on local context surrounding each table table. In the second phase, the embeddings for the query and candidate tables are updated while incorporating context about the other table's columns.

4.3.1 Update Phase 1: Phase 1 updates the embeddings of entities in the metadata KG for the query and candidate tables independently from each other. Given the sparsity of connections among entities in the metadata KG, we find that collecting only the 2-hop neighborhood is sufficient – this allows the subgraph to encompass all metadata for a given table, other tables which share the same tags, and other tables and their columns which are joinable with the table.

For the subgraphs G_Q and G_C, we update the representations of all entities using a GNN model. Here, we construct a heterogeneous GNN of depth 2 to perform the forward passes. By using a heterogeneous GNN structure, different GNN layer weights will be applied to each relation type (e.g., a tag and column will have a different influence on updating the representation of a table entity). In principle, any GNN architecture can be used in this part of our model. In our experiments, we choose to utilize GATv2 [4] layers, given their strong performance and ease of use.

Using this model we produce updated embeddings for $\vec{t_Q}'$ and $\vec{t_C}'$, as well as their respective columns, $\vec{c_i}' \in C_Q$ and $\vec{c_j}' \in C_C$.An example of how updated embeddings are produced from this model is shown in Figure 3 – note that layer details are greatly simplified, and many neighboring entities for t_Q are omitted for brevity. At each depth, the GNN processes and aggregates information from neighboring nodes, allowing the output embeddings to be influenced by entities up to 2-hops away.

4.3.2 Update Phase 2: For the second phase of MeGNN-Join, we aim to produce updated table representations for the query table which incorporates context about the candidate table's columns, and vice versa. Intuitively, we expect that allowing each column in one table to influence columns in the other table will help to further contextualize the meaning of each column and understand whether the columns represent joinable concepts or not.

For each column entity $c_i \in C_Q$ and $c_j \in C_C$, we augment a new relation from every c_i to every c_j and vice versa to our subgraphs G_Q and G_C. A visualization of this step can be seen in the "Add column connections" step of Figure 1, where we essentially add edges among all columns of the two tables. These additional relations can now be used by a second GNN model to once again produce updated embeddings for t_Q and t_C, in a similar fashion as our example in Figure 3.

With the augmented subgraphs, the first depth of the GNN allows each column entity to incorporate information from other table's columns, and the second depth allows these updated column vectors to then influence t_Q and t_C, producing the vectors $\vec{t_Q}''$ and $\vec{t_C}''$ as the tables' final representations.

4.3.3 Joinability Score: The joinability score is produced by applying a sigmoid function to the dot product of $\vec{t_Q}''$ and $\vec{t_C}''$. This score is essentially used as a classifier, classifying the tables as joinable or not, and we use this to compute binary cross entropy loss to train our model. Through training MeGNN-Join, tables that are joinable should have similar final representations and tables that are not should be dissimilar.

5 EVALUATION FRAMEWORK

Given the limited availability of datasets of tabular data with associated metadata, we opt to devise a new framework to evaluate our work. We curate three datasets of real-world tabular data, and produce two variations of joinability data for each. We additionally construct one synthetic dataset based on YAGO 4.5 [33]. We then design experiments for three different evaluations settings.

5.1 Data Curation

We curate three datasets using metadata APIs offered by open government data portals: **CONY**, from the NYC Open Data portal; **NYS**, from the Open NY portal; and **CANADA**, from Canada.ca's open government portal. Each data portal provided varying amounts of metadata – CONY contained the most metadata, while NYS contained no row labels, and CANADA contained no row labels or column descriptions. We also performed filtering over the tables to remove (1) tables with fewer than 10 rows, to remove cases where the tiny number of rows would heavily bias the containment ratio calculation in our ground truth generation (described in Section 5.2), and (2) tables containing personally identifiable information, to avoid any potential privacy concerns with the released dataset. After performing filtering, the total number of tables for CONY, NYS, and CANADA were **2,049**, **714**, and **1,237**.

5.2 Ground-Truth Generation

The main goal of our evaluation is to determine how the joinability predictions produced by our approach, which uses only table metadata, compares to table joins that a system produces when it has access to table contents. Therefore, we generate joinability data using content-based methods to serve as our ground-truth for these three datasets.

For the ground-truth joins, we use an open source implementation[2] to compute the set similarity of column contents. In particular, we use the set containment ratio as our key metric – for a pair of columns and their corresponding sets of cell values A

[2]https://github.com/ekzhu/setsimilaritysearch

and B, the containment ratio is calculated as $\frac{|A \bigcap B|}{min(|A|,|B|)}$. Having a high containment ratio indicates that the columns contain a greater proportion of cell values on which the tables can be joined.

To determine joinable tables using this metric, we first filter out columns containing numeric data, as it gave us many false positives for joinable columns. Next, we filter out columns where less than 10% of the cells had unique values so that we can ignore cases where the columns contain uninteresting values (like columns indicating "yes"/"no") and avoid the containment ratio being heavily biased by columns with few unique values. We then compute the containment ratio among all columns, and tables containing columns with a ratio above a set threshold were judged to be joinable. In our evaluation, we produce two versions of the joinability data with thresholds of 0.6 and 0.3. Experiments using a threshold value of 0.6 indicates a stricter joinability criteria, as at least one column must share more than half of its cell values with the other column. The total number of joins produced for the threshold of 0.6 for the CONY, NYS, and CANADA datasets were **11,422**, **2,050**, and **1,024**, respectively. For the threshold of 0.3, the number of joins for the three datasets were **28,396**, **4,920**, and **3,166**.

Our final dataset, **YAGO**, was constructed synthetically by considering classes in the YAGO 4.5 KG as "tables", instances as "rows", and properties as "columns". Intuitively, we consider instance entities that share the same property and tail entity as being joinable on that particular property, and generate our synthetic dataset to reflect such joins. We first selected all entities which were instances of exactly one Class, then collected classes that contained at least 5 instances which were joinable with another class to form our final dataset. YAGO consists of **880** tables with **59,892** joins. Unlike our other datasets, in YAGO all "columns" that share the same name represent the same concept; however, this does not necessarily mean that they are joinable unless the columns also share some values.

Table 1: Summary statistics for our curated datasets.

	CONY	NYS	CANADA	YAGO
Tables	2,049	714	1,237	880
Descriptions	1,953	714	1,237	763
Columns	40,986	10,554	28,546	3,444
Column Descriptions	9,112	9,402	0	3,444
Row Labels	549	0	0	0
Tags	2,606	1,496	3,516	0

Table 1 shows an overview of the general dataset statistics for our curated datasets. We note that each dataset has varying amounts of available metadata, with only CONY having row label metadata and CANADA not having any column descriptions. Having this variety of metadata will help to give us insight into the robustness of our method, and also presents the opportunity to explore the impact of automatically generated metadata enrichments.

5.3 Evaluation Settings

We assess model performance following standard procedures for link prediction in KG research. For each test sample, joinability predictions are made by producing a joinability score for all other tables and ranking them. We then calculate filtered mean reciprocal rank (MRR) – i.e., the average of $\frac{1}{rank}$, where the correct prediction occurs at $rank$ – and Hits@10 – the proportion of test samples where the correct prediction occurs within the top 10 ranks – as our main evaluation metrics. We also consider the three evaluation settings; transductive, semi-inductive, and inductive.

In the **transductive** setting, which is typically used in evaluating link prediction, we assume that all tables in the test set are also seen during training. Data is split into train and test sets by selecting a set of table join *edges* to set aside for testing. For the **inductive** setting, we perform predictions for tables that were not seen during training. Data in this setting is split by selecting a set of *tables* to set aside for testing, and joins are predicted among the test tables. Lastly, the **semi-inductive** setting aims to predict joins between a pair of tables where one was seen during training and one was not. This setting uses the same data split as the inductive setting, but join predictions are performed only for join pairs where one table is from the test set and one is from the train set.

To contextualize each of these evaluation settings: the transductive setting corresponds to a situation where we have a data lake which already contains table join information, and we wish to enrich it further by discovering missing joins; the semi-inductive setting is where we have a data lake with join information, and we wish to introduce a new table and identify what tables it is joinable with; and the inductive setting serves to evaluate how training a model on an existing set of tables in a data lake can be applied to identify joins among a whole new set of tables from scratch.

Experiments were performed for each dataset and setting using 5-fold cross validation; detailed data split statistics can be found in Section 2 of our Supplemental Material.

5.4 Difficult Joins

Another factor which we will analyze in our experiments is the performance of various models in situations where it might be *easy* to determine the joinability of tables versus *difficult* joins. Here, we consider any two tables that are joinable on columns with different names as difficult (e.g., joining the "Community Board" column of one table to the "CD Name" column of another). Easy joins can plausibly be identified using exact column name matching (which we include as a baseline), while difficult joins cannot.

5.5 Models and Baselines

To the best of our knowledge, no recent work exists surrounding the task of predicting table joinability using only table metadata. To compare against our model, we use several text-embedding based baselines over metadata features as well as common link prediction methods. Further details about baseline models and experiment configurations are provided in Section 3 of our Supplemental Material.

Text Embeddings: We consider three text-based baselines. The first, Exact Column, scores table joinability by counting the number of columns in each table with exactly-matching names. The second, ColumnSim, uses SBERT embeddings of column names and computes a weighted sum of their cosine similarities to produce a score. The third is TableSim, which scores joinability based on the cosine similarity of table names' SBERT embeddings. ColumnSim and TableSim are comparable to how a state of the art content-based

Table 2: Evaluation results, with best results highlighted in bold.

	Transductive			Semi-inductive			Inductive		
Method	MRR	Hits@10	Rank	MRR	Hits@10	Rank	MRR	Hits@10	Rank
Exact Column	.204	.288	11.6	.263	.390	5.71	.305	.455	4.71
ColumnSim	.188	.327	11.0	.262	.417	5.86	.320	.471	5.00
TableSim	.188	.315	11.0	.264	.432	6.29	.309	.520	4.71
TransE	.726	.872	4.29	-	-	-	-	-	-
DistMult	.682	.873	5.86	-	-	-	-	-	-
ComplEx	.718	.902	4.57	-	-	-	-	-	-
Rotate	.468	.628	8.14	-	-	-	-	-	-
GraIL	.292	.538	9.14	.080	.193	8.00	.077	.225	8.00
AnyBURL	.754	.906	3.57	.449	.575	3.57	.210	.407	6.86
GNN	.605	.831	6.57	.506	.706	3.14	.458	.646	2.43
MeGNN-Join (Ours)	.831	**.952**	1.71	.591	.751	2.14	**.500**	**.679**	**1.43**
MeGNN-Join$_J$ (Ours)	**.836**	.951	**1.57**	**.671**	**.842**	**1.00**	.427	.616	2.71

method such as DeepJoin [8] could be applied to our setting, under the constraint that no cell values are available. We apply the same SBERT model[3] as the one used in our MeGNN-Join implementation for these baselines.

KG Embeddings: As our task setup is one of link prediction, we also include several popular baseline link prediction methods which are based on embedding the entities and relations of the KG into a vector space. Among the many KG embedding models [13, 38, 39], we chose four widely used models; TransE [3], ComplEx [36], DistMult [40], and RotatE [34]. These models are only evaluated in the transductive setting, as they do not support inductive predictions.

Rule Learning: Another class of models that can perform link prediction in KGs are those which aim to learn rules or patterns from the KG [19, 23, 24]. We use AnyBURL [23] as a baseline for rule learning given its strong performance and efficiency, as well as its ease of applicability to the inductive setting.

GNNs: We include two GNN-based baselines which are capable of performing inductive link prediction. The first is GraIL [35], which can predict links between entities based only on the local subgraph structure. We indicate the second baseline simply as GNN. This baseline mirrors the modeling and structure of MeGNN-Join, using GATv2 as the GNN layers, but omits our second update phase, demonstrating how a normal GNN model would be applied to our core idea of using a metadata KG with text embeddings.

MeGNN-Join: We evaluate two variations of our model – one that incorporates other table joins (present in the training set) in its training and inference, which we denote as MeGNN-Join$_J$, and one that does not incorporate such joins into training or inference, as MeGNN-Join. We expect MeGNN-Join$_J$ to be able to take greater advantage of existing knowledge about tables seen during training, at the cost of generalizability when predicting joins for new tables.

6 RESULTS

Experiments were performed for each dataset and evaluation setting using 5 fold cross validation. In the following section we highlight average results across each cross validation and datasets. Detailed evaluation results for each dataset are available in Section 4 of our Supplemental Material.

Our main experiment's results are shown in Table 2, where we present the average MRR↑, Hits@10↑, and the relative Rank↓ of each model (where each model is ranked based on its MRR in each cross validation split and dataset) for each evaluation setting. MeGNN-Join$_J$ shows the strongest performance in the transductive and semi-inductive settings, while MeGNN-Join without the additional join information is stronger in the inductive setting. Among the baselines, we observe that typical link prediction models are able to show reasonable performance in the transductive setting, which supports the idea that link prediction for this sort of table join prediction is a viable approach. However, in the more difficult semi-inductive and inductive settings, other baselines begin to struggle more. MeGNN-Join demonstrates robust performance when applied to tables that were unseen during training. We also observe that MeGNN-Join consistently outperform the standard GNN baseline, indicating the benefits of our secondary update phase as well as semantic enrichment procedure.

Table 3: Average model performance for difficult joins.

	Transductive	Semi-inductive	Inductive
Method	Hits@10	Hits@10	Hits@10
ColumnSim	.018	.045	.052
TableSim	.064	.128	.136
AnyBURL	.847	.320	.020
GNN	.725	.663	.336
MeGNN-Join	.913	.737	**.369**
MeGNN-Join$_J$	**.920**	**.818**	.261

6.1 Difficult Joins

Table 3 shows the evaluation results in terms of Hits@10, focusing on the two most competitive baselines (AnyBURL and GNN) as well as two text embedding baselines. Detailed results for difficult joins can be found in Section 5 of our Supplemental Material.

[3] huggingface.co/sentence-transformers/all-MiniLM-L6-v2

We see that using the text embeddings of table or column names alone performs extremely poorly across all evaluation settings for these difficult joins. AnyBURL remains competitive in the transductive setting, but its performance heavily degrades in the semi-inductive and inductive setting because many of its learned rules rely on entities that were seen during training or joinability links among other tables. MeGNN-Join shows robust performance in the transductive and semi-inductive setting, although it does begin to struggle in the inductive setting, and the performance gap between MeGNN-Join and the standard GNN becomes quite small.

6.2 Ablation Study

To gain insight into the effects of various pieces of metadata, as well as to validate the use of our automatically generated metadata, we perform an ablation study to explore changes in model performance.

Table 4 shows the performance of MeGNN-Join$_J$ in the transductive setting on our three open data portal datasets, comparing the MRR performance when the model is trained and tested end-to-end with or without the inclusion of automatically enriched metadata (column name expansions, column descriptions, and keywords). We can observe that the enriched metadata is often useful – while the semantic enrichment has no impact on the performance of NYS, for the CONY dataset the enrichments provide roughly a 7% improvement in performance. Note that in enterprise data lake settings, enrichment is likely even more impactful due to sparser metadata in enterprise data catalogs.

Table 4: Comparison of model performance with and without enrichments.

Dataset	Enriched	MRR
CONY	Yes	.833
CONY	No	.784
NYS	Yes	.766
NYS	No	.767
CANADA	Yes	.753
CANADA	No	.744

Table 5: Changes in model performance when specific pieces of metadata are removed from the KG.

Removed Metadata	MRR
None	0.833
Semantic Enrichments	0.784
Column descriptions	0.776
Tags	0.761
Row labels	0.758
Table descriptions	0.684

Table 5 investigates the impact of removing a specific piece of metadata from the metadata KG, making it unavailable during inference. This ablation study is conducted for the CONY dataset in the transductive setting. While MeGNN-Join$_J$ is robust in being able to produce inference results even if portions of the metadata are missing, we can observe that removing any piece of metadata negatively affects performance. This underscores how all parts of the metadata provide valuable contributions to the model.

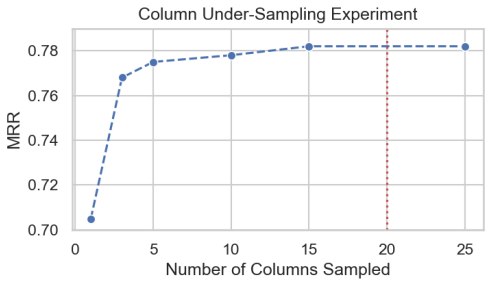

Figure 4: Ablation study on under-sampling columns for CONY-C tables.

We also consider an additional ablation study regarding *how much* column information is available from each table during inference. Here, we only sample a subset of the tables' columns before processing it with MeGNN-Join and producing a prediction. Each experiment is repeated 10 times and we report the average MRR – results can be seen in Figure 4. Interestingly, we observe that even sampling a single column leads to an MRR of 0.705, and by the time we sample 10 columns our model nearly reaches its full performance, despite the fact that tables in CONY-C on average have 20 columns. This further suggests how our method is capable of robustly utilizing metadata, even when large portions of it are missing.

7 CONCLUSION

We present MeGNN-Join, an approach for predicting joinability among tabular datasets using only their metadata. MeGNN-Join operates by framing this task as a link prediction task by forming a KG from the tables' metadata, and we introduce a pipeline to leverage text embeddings together with GNNs to produce table representations. Additionally, we introduce a semantic enrichment step to automatically generate missing pieces of metadata, which might be crucial in many real-world datasets where high-quality metadata is sparse. We evaluate MeGNN-Join's ability to reproduce content-based joinability results using only metadata, and demonstrate strong performance against baselines, especially in more difficult evaluation settings. Our results show that a metadata-based approach for table joinability offers a compelling and practical direction for future research. It enables lightweight and privacy-preserving use of data, particularly in settings where direct access to table contents is restricted. Moreover, it opens up new opportunities for scalable interaction with data on the web. In future work, we hope to expand upon this research in several directions such as exploring more diverse dataset domains, incorporating semantic table interpretation solutions [16, 18, 43], and deeper investigations into the use and evaluation of automatic metadata enrichment.

REFERENCES

[1] Jinze Bai, Jialin Wang, Zhao Li, Donghui Ding, Ji Zhang, and Jun Gao. 2021. ATJ-Net: Auto-Table-Join Network for Automatic Learning on Relational Databases.

In *Proceedings of the Web Conference 2021 (WWW '21)*. Association for Computing Machinery, New York, NY, USA, 1540–1551. https://doi.org/10.1145/3442381.3449980

[2] Sagar Bharadwaj, Praveen Gupta, Ranjita Bhagwan, and Saikat Guha. 2021. Discovering related data at scale. *Proceedings of the VLDB Endowment* 14, 8 (2021), 1392–1400.

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-Relational Data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2* (Lake Tahoe, Nevada) *(NIPS'13)*. Curran Associates Inc., Red Hook, NY, USA, 2787–2795.

[4] Shaked Brody, Uri Alon, and Eran Yahav. 2021. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491* (2021).

[5] Raul Castro Fernandez, Ziawasch Abedjan, Famien Koko, Gina Yuan, Samuel Madden, and Michael Stonebraker. 2018. Aurum: A Data Discovery System. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. 1001–1012. https://doi.org/10.1109/ICDE.2018.00094 ISSN: 2375-026X.

[6] Li Ding, Dominic DiFranzo, Alvaro Graves, James Michaelis, Xian Li, Deborah L. McGuinness, and Jim Hendler. 2010. Data-gov Wiki: Towards Linking Government Data. In *Linked Data Meets Artificial Intelligence, Papers from the 2010 AAAI Spring Symposium, Tech. Report SS-10-07*. AAAI.

[7] Yuyang Dong, Kunihiro Takeoka, Chuan Xiao, and Masafumi Oyamada. 2021. Efficient Joinable Table Discovery in Data Lakes: A High-Dimensional Similarity-Based Approach. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. 456–467. https://doi.org/10.1109/ICDE51399.2021.00046

[8] Yuyang Dong, Chuan Xiao, Takuma Nozawa, Masafumi Enomoto, and Masafumi Oyamada. 2023. DeepJoin: Joinable Table Discovery with Pre-Trained Language Models. *Proc. VLDB Endow.* 16, 10 (jun 2023), 2458–2470. https://doi.org/10.14778/3603581.3603587

[9] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783* (2024).

[10] Jason Ellis, Achille Fokoue, Oktie Hassanzadeh, Anastasios Kementsietsidis, Kavitha Srinivas, and Michael J. Ward. 2015. Exploring Big Data with Helix: Finding Needles in a Big Haystack. *SIGMOD Rec.* 43, 4 (2015), 43–54. https://doi.org/10.1145/2737817.2737829

[11] Grace Fan, Jin Wang, Yuliang Li, and Renée J. Miller. 2023. Table Discovery in Data Lakes: State-of-the-art and Future Directions. In *Companion of the 2023 International Conference on Management of Data (SIGMOD '23)*. Association for Computing Machinery, New York, NY, USA, 69–75. https://doi.org/10.1145/3555041.3589409

[12] Grace Fan, Jin Wang, Yuliang Li, Dan Zhang, and Renée J. Miller. 2023. Semantics-Aware Dataset Discovery from Data Lakes with Contextualized Column-Based Representation Learning. *Proceedings of the VLDB Endowment* 16, 7 (March 2023), 1726–1739. https://doi.org/10.14778/3587136.3587146

[13] Xiou Ge, Yun Cheng Wang, Bin Wang, C-C Jay Kuo, et al. 2024. Knowledge Graph Embedding: An Overview. *APSIPA Transactions on Signal and Information Processing* 13, 1 (2024).

[14] Yeye He, Kris Ganjam, and Xu Chu. 2015. SEMA-JOIN: joining semantically-related tables using big table corpora. *Proceedings of the VLDB Endowment* 8, 12 (Aug. 2015), 1358–1369. https://doi.org/10.14778/2824032.2824036

[15] Ahmed Helal, Mossad Helali, Khaled Ammar, and Essam Mansour. 2021. A Demonstration of KGLac: A Data Discovery and Enrichment Platform for Data Science. *Proc. VLDB Endow.* 14, 12 (jul 2021), 2675–2678. https://doi.org/10.14778/3476311.3476317

[16] Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, and Kavitha Srinivas. 2020. SemTab 2019: Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems. *European Semantic Web Conf. (ESWC '19)* 12123 (2020), 514 – 530.

[17] Myung Jun Kim, Léo Grinsztajn, and Gaël Varoquaux. 2024. CARTE: pretraining and transfer for tabular learning. *arXiv preprint arXiv:2402.16785* (2024).

[18] Craig A Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyan, and Parag Mallick. 2012. Semi-automatically mapping structured sources into the semantic web. In *Extended semantic web conference*. 375–390.

[19] Jonathan Lajus, Luis Galárraga, and Fabian Suchanek. 2020. Fast and Exact Rule Mining with AMIE 3. In *The Semantic Web*. Springer International Publishing, Cham, 36–52. https://doi.org/10.1007/978-3-030-49461-2_3

[20] Oliver Lehmberg, Dominique Ritze, Petar Ristoski, Robert Meusel, Heiko Paulheim, and Christian Bizer. 2015. The Mannheim Search Join Engine. *Journal of Web Semantics* 35 (Dec. 2015), 159–166. https://doi.org/10.1016/j.websem.2015.05.001

[21] Guoliang Li, Jian He, Dong Deng, and Jian Li. 2015. Efficient Similarity Join and Search on Multi-Attribute Data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD '15)*. Association for Computing Machinery, New York, NY, USA, 1137–1151. https://doi.org/10.1145/2723372.2723733

[22] Elita Lobo, Oktie Hassanzadeh, Nhan Pham, Nandana Mihindukulasooriya, Dharmashankar Subramanian, and Horst Samulowitz. 2023. Matching Table Metadata with Business Glossaries Using Large Language Models. *arXiv preprint arXiv:2309.11506* (2023).

[23] Christian Meilicke, Melisachew Chekol, Manuel Fink, and Heiner Stuckeschmidt. 2023. Anytime bottom-up rule learning for large-scale knowledge graph completion. *The VLDB Journal* (06 2023), 1–31. https://doi.org/10.1007/s00778-023-00800-5

[24] Christian Meilicke, Manuel Fink, Yanjie Wang, Daniel Ruffinelli, Rainer Gemulla, and Heiner Stuckenschmidt. 2018. Fine-Grained Evaluation of Rule- and Embedding-Based Systems for Knowledge Graph Completion. In *ISWC*, Vol. 11136. Springer, 3–20. https://doi.org/10.1007/978-3-030-00671-6_1

[25] Nandana Mihindukulasooriya, Sarthak Dash, Sugato Bagchi, Ariel Farkash, Igor Gokhman, Oktie Hassanzadeh, Nhan Pham, et al. 2023. Unleashing the Potential of Data Lakes with Semantic Enrichment Using Foundation Models.. In *ISWC (Posters/Demos/Industry)*.

[26] Renée J. Miller. 2018. Open data integration. *Proceedings of the VLDB Endowment* 11, 12 (Aug. 2018), 2130–2139. https://doi.org/10.14778/3229863.3240491

[27] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Conference on Empirical Methods in Natural Language Processing*. https://api.semanticscholar.org/CorpusID:201646309

[28] Aécio Santos, Aline Bessa, Christopher Musco, and Juliana Freire. 2022. A Sketch-based Index for Correlated Dataset Search. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. 2928–2941. https://doi.org/10.1109/ICDE53745.2022.00264

[29] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The Graph Neural Network Model. *IEEE Transactions on Neural Networks* 20, 1 (2009), 61–80. https://doi.org/10.1109/TNN.2008.2005605

[30] Juan Sequeda, Dean Allemang, and Bryon Jacob. 2023. A Benchmark to Understand the Role of Knowledge Graphs on Large Language Model's Accuracy for Question Answering on Enterprise SQL Databases. *CoRR* abs/2311.07509 (2023). https://doi.org/10.48550/ARXIV.2311.07509 arXiv:2311.07509

[31] Divesh Srivastava. 2010. Schema extraction. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*. ACM, 3–4. https://doi.org/10.1145/1871437.1871440

[32] Pranav Subramaniam, Udayan Khurana, Kavitha Srinivas, and Horst Samulowitz. 2023. NumJoin: Discovering Numeric Joinable Tables with Semantically Related Columns. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23)*. Association for Computing Machinery, New York, NY, USA, 5096–5100. https://doi.org/10.1145/3583780.3614750

[33] Fabian M. Suchanek, Mehwish Alam, Thomas Bonald, Pierre-Henri Paris, and Jules Soria. 2023. YAGO 4.5: A Large and Clean Knowledge Base with a Rich Taxonomy. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[34] Zhiqing Sun, Zhihong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. *ICLR* (2019).

[35] Komal K. Teru, E. Denis, and William L. Hamilton. 2019. Inductive Relation Prediction by Subgraph Reasoning. In *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:211082667

[36] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings)*, Vol. 48. JMLR.org, 2071–2080.

[37] Sebastian Wandelt, Dong Deng, Stefan Gerdjikov, Shashwat Mishra, Petar Mitankin, Manish Patil, Enrico Siragusa, Alexander Tiskin, Wei Wang, Jiaying Wang, and Ulf Leser. 2014. State-of-the-art in string similarity search and join. *ACM SIGMOD Record* 43, 1 (May 2014), 64–76. https://doi.org/10.1145/2627692.2627706

[38] Meihong Wang, Linling Qiu, and Xiaoli Wang. 2021. A Survey on Knowledge Graph Embeddings for Link Prediction. *Symmetry* 13, 3 (2021). https://doi.org/10.3390/sym13030485

[39] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743. https://doi.org/10.1109/TKDE.2017.2754499

[40] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).

[41] Minghe Yu, Guoliang Li, Dong Deng, and Jianhua Feng. 2016. String similarity search and join: a survey. *Frontiers of Computer Science* 10, 3 (June 2016), 399–417. https://doi.org/10.1007/s11704-015-5900-5

[42] Jiani Zhang, Zhengyuan Shen, Balasubramaniam Srinivasan, Shen Wang, Huzefa Rangwala, and George Karypis. 2023. NameGuess: Column Name Expansion for Tabular Data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 13276–13290. https:

//doi.org/10.18653/v1/2023.emnlp-main.820
[43] Ziqi Zhang. 2014. Towards efficient and effective semantic table interpretation. In *ISWC*. 487–502.
[44] Erkang Zhu, Dong Deng, Fatemeh Nargesian, and Renée J. Miller. 2019. JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes. In *Proceedings of the 2019 International Conference on Management of Data (SIGMOD '19)*. Association for Computing Machinery, New York, NY, USA, 847–864. https://doi.org/10.1145/3299869.3300065