OpenForge: Probabilistic Metadata Integration

Tianji Cong
University of Michigan
congtj@umich.edu

Fatemeh Nargesian
University of Rochester
fnargesian@rochester.edu

Junjie Xing
University of Michigan
jjxing@umich.edu

H. V. Jagadish
University of Michigan
jag@umich.edu

ABSTRACT

Modern data stores increasingly rely on metadata to enable diverse activities such as data cataloging and search. However, metadata curation remains a labor-intensive task, and the broader challenge of metadata maintenance—ensuring its consistency and usefulness—has been largely overlooked. In this work, we tackle the problem of resolving relationships among metadata concepts from disparate sources. Inferring these relationships are critical for creating clean and consistent metadata repositories, and a central challenge for metadata integration.

We propose OpenForge, a two-stage prior-posterior framework for metadata integration. In the first stage, OpenForge exploits multiple methods including fine-tuned large language models to obtain prior beliefs about concept relationships. In the second stage, OpenForge refines these predictions using the Markov Random Field, a probabilistic graphical model. We formalize metadata integration as an optimization problem, where the objective is to identify the relationship assignments that maximize the joint probability of assignments. The MRF formulation allows OpenForge to capture prior beliefs while encoding critical relationship properties, such as transitivity, in probabilistic inference. Experiments on four datasets show the effectiveness and efficiency of OpenForge. In a use case of matching two metadata vocabularies, OpenForge outperforms GPT-4, the second-best method, by 25 F1 points.

PVLDB Reference Format:
Tianji Cong, Fatemeh Nargesian, Junjie Xing, and H. V. Jagadish. OpenForge: Probabilistic Metadata Integration. PVLDB, 18(9): 2914 - 2927, 2025.
doi:10.14778/3746405.3746417

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/superctj/openforge.

1 INTRODUCTION

Clean and consistent metadata is pivotal in enabling the FAIR Data Principles–findability, accessibility, interoperability and reusability–in data repositories [37, 42, 60]. While significant progress has been made in data discovery through techniques that leverage data values [5, 8, 10, 13, 22, 41, 43], real-world dataset search engines [8, 26, 56] rely heavily on metadata (e.g., keywords) for indexing and retrieval. However, the practice of metadata curation, for dataset publishing and indexing, remains labor-intensive [57]. Ensuring metadata consistency across disparate dataset sources is an ongoing challenge, especially as datasets grow in scale and diversity [57].

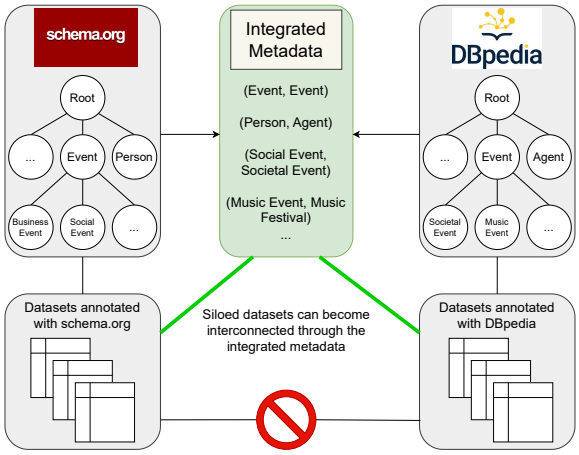

Figure 1: Illustration of metadata integration problem.

Metadata Granularity and Vocabulary Mismatch Most repositories rely heavily on metadata, on which they place significant requirements, including the vocabulary used. For example, Google dataset search engine [8] requires data publishers to annotate their datasets using the `schema.org` vocabulary; otherwise, their datasets would not be well indexed for search. Web Data Commons [12], a longstanding effort in curating web tables, publishes benchmarks [30] annotated with both Schema.org types and DBpedia classes [33]. In contrast, open data portals like `CKAN` [11] and `Socrata` [55] allow metadata to be provided in JSON format with open vocabulary fields (e.g., `description`, `publisher`, `theme`). As a result, metadata granularity and vocabulary–ranging from controlled schemas to open-ended terms–vary widely across datasets, organizations, and repositories.

Practice of Metadata Curation A close examination of curatorial work at ICPSR, the world's largest social science data archive [26], reveals that (meta)data curation involves domain expertise and subjective judgments to resolve inconsistencies and ensure standardization [57]. Curators revise dataset descriptions and create metadata records, such as subject terms and geographic coverage, to improve dataset search. Such tasks are often guided

Proceedings of the VLDB Endowment, Vol. 18, No. 9 ISSN 2150-8097.
doi:10.14778/3746405.3746417

by metadata standards, like a manually curated vocabulary or thesaurus. However, experienced curators view such standards as flexible rather than rigid [57]. To remain relevant, curated vocabularies—and the metadata based on them—must evolve alongside changing data landscapes. Similarly, curators of platforms like `data.gov` must reconcile diverse metadata standards, as individual states and cities independently collect and manage their datasets.

Maintaining consistent and useful metadata, both within and across repositories, requires developing techniques to (1) unify and standardize metadata elements, (2) refine their granularity, and (3) perform holistic maintenance at scale. Motivated by these challenges, we study the problem of integrating metadata from disparate sources to build a consistent and granular view of metadata, by unifying equivalent metadata elements and defining metadata semantic hierarchies. Such hierarchies can be constructed using two primary types of relationships between metadata elements–*equivalence* and *parent-child*–as established in knowledge base literature. As illustrated in Figure 1, given a collection of potentially noisy metadata elements and associated data from various sources (optionally including dataset contents and descriptions), our goal is to identify all equivalence and parent-child relationships among element pairs.

We propose OpenForge, a data-driven framework that integrates and resolves inconsistencies among a collection of metadata elements. Conceptually, OpenForge's output can be modeled as a graph of metadata elements (concepts) linked with two types of relationships. We call this graph, a *relationship assignment graph*. Since the relationship instantiations are not known apriori, we cast the problem of constructing this graph as a probabilistic inference problem. At the core of our approach lies a unified probabilistic formulation based on Markov Random Field (MRF), an undirected graphical model, which can integrate various types of relationships within a single framework. This formulation enables us to (1) incorporate diverse methods that leverage metadata elements and accompanying data (if available) to computing prior probabilities, (2) encode relationship axioms, such as transitivity, to prevent conflicting relationship instantiations and infer the most probable relationship assignment.

Our model represents relationships between elements as random variables within a MRF, framing relationship assignment as a joint probability distribution over these variables. The optimal relationship graph is then the assignment that maximizes this joint probability. By definition, MRFs decompose the joint probability into a product of factors which are non-negative functions defined over cliques of the graph. In particular, we leverage ternary factors (factors defined over ternary cliques of random variables) to encode transitivity where invalid configurations of random variables are assigned a probability of zero. This structural enforcement of transitivity is a key reason for adopting the MRF framework, alongside its principled handling of joint dependencies. However, the challenge lies in tuning the relative probabilities among valid (transitive) configurations, which critically influence model behavior. Manually setting these parameters risks poor generalization across datasets. To address this, we employ Bayesian optimization to learn optimal parameters in a data-driven manner, as described in Section 5.2. OpenForge also flexibly integrates a wide range of prior models–including prompting and fine-tuning large language models (LLMs), as well as traditional machine learning models–allowing the MRF to combine learned priors with structured inference.

Another key challenge in applying probabilistic graphical models to real-world data is inference scalability [1, 59, 64]. This challenge is exacerbated for OpenForge when operating on large datasets with sparse relationships. OpenForge addresses this challenge through an embedding-based strategy that decomposes the MRF into smaller, independent subgraphs based on metadata semantics, enabling efficient parallelized inference over these subgraphs.

In summary, we make the following contributions:

- We introduce the metadata integration problem as crucial for unification and integrity maintenance of metadata repositories, identifying equivalence and parent-child relationship determination as central tasks to address this problem (Section 2.1).
- We propose a probabilistic formulation for resolving equivalence and parent-child relationships among metadata elements (Section 2.2), framing the task as maximum a posteriori inference in a Markov Random Field (Section 3.2).
- We develop OpenForge, a data-driven technique that integrates various prior models, including LLMs, and refines their predictions through probabilistic inference (Section 3).
- To address the scalability challenge of probabilistic inference on large sparse MRFs, we design an embedding-based strategy that decomposes sparse MRFs into smaller independent subgraphs, enabling parallelized inference for large datasets (Section 4).
- Experiments on four real-world datasets show that OpenForge outperforms task-specific baselines and the latest LLMs (including GPT-4) significantly. We also demonstrate efficient MRF inference completes efficiently on the largest dataset and scalability to synthetic graphs with millions of random variables (Section 6).

2 METADATA INTEGRATION PROBLEM

2.1 Problem Statement

A metadata repository is the union of the metadata of a collection of datasets. To formalize a metadata repository, we consider metadata elements and their relationships as the building blocks of a metadata repository. Suppose a set of elements $\mathcal{V} = \{c_1, c_2, \ldots, c_n\}$, where each element represents a unit of metadata information for integration such as column annotations, keywords, entity mentions, or dataset metadata attributes. We call each element in $\mathcal{V}$ a metadata element or concept, interchangeably. Consider the relationship $R : \mathcal{V} \times \mathcal{V} \rightarrow \{0, 1\}$ that can hold between any pair of concepts, e.g., equivalence or parent-child. Our concrete goal is to enrich the metadata of a data repository with a given relationship R. More specifically, we want to infer the presence or absence of R between each pair of concepts in $\mathcal{V}$. The concepts in $\mathcal{V}$ are raw metadata units extracted from metadata and dataset files. However, discovering equivalence relationships between pairs of concepts will allow us to unify and standardize these raw concepts. Note that data repositories may contain various types of metadata (e.g., `keyword`, `theme`, `column annotation`). A data curator can choose to integrate specific types or include all metadata, defining $\mathcal{V}$ accordingly. In addition to metadata elements in $\mathcal{V}$, data repositories may also include accompanying data content or descriptions. We refer to this accompanying information, if present, as *evidence*, as described in Section 2.2.

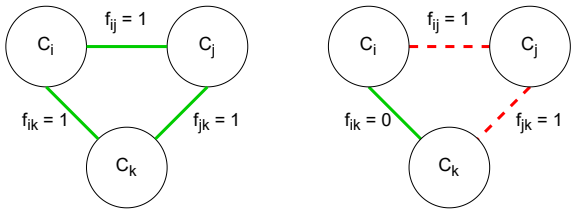

Figure 2: Illustration of the relationship transitivity (left) and inconsistent relationship assignments that violate the transitivity (right). Green solid edges indicate correct predictions and red dashed edges indicate conflicting predictions.

In this paper, we focus on two types of relationships: equivalence and parent-child relationships. We note that identifying equivalent elements and parent-child elements is similar to tasks in data matching [58] and taxonomy induction [54], respectively. Although these two relationships are typically addressed with task-specific methods [14, 34, 49, 58, 62, 63], we demonstrate below that they can be effectively modeled within the same probabilistic formulation. This unified approach reduces complexity and maintenance overhead compared to using separate methods, and our experiments on four tasks empirically validate our approach's ability to handle both relationship types.

2.2 Probabilistic Formulation

We first define a relationship assignment graph for relationship R, namely G_R, where each node represents a metadata concept in $\mathcal{V}$. The edges in G_R encode whether and how R holds between pairs of concepts. Specifically, for any two distinct concepts c_i and c_j, we define a random variable r_{ij} that characterizes the relationship:

- If R is bi-directional, like equivalence, r_{ij} is a binary random variable taking values in $\{0, 1\}$ where $r_{ij} = 1$ indicates that the relationship is present in both directions and $r_{ij} = 0$ means that no relationship exists.
- If R is uni-directional, like parent-child, r_{ij} is a categorical random variable taking values in $\{0, 1, 2\}$ where $r_{ij} = 1$ indicates that the relationship is directed from c_i to c_j, $r_{ij} = 2$ indicates that the relationship is directed from c_j to c_i, and $r_{ij} = 0$ means that there is no relationship.

The relationship assignment graph G_R can therefore be undirected or directed, depending on the nature of R. In both cases, each random variable follows a discrete probability distribution. When it is clear from the context, we refer to G_R simply as G.

Our goal is to find the most probable relationship assignment graph G defined over $\mathcal{V}$. We define the probability of any given relationship assignment graph G as the joint probability of all random variables r_{ij} formed for $\mathcal{V}$ of G, capturing both the presence and, if applicable, the directionality of relationships.

Definition 1 (Probability of a Relationship Assignment Graph)**.**

$$P(G) = P(\{ r_{ij} \mid (c_i, c_j) \in \mathcal{V} \times \mathcal{V}, i < j \}).$$

We consider only one ordering of concept pairs, meaning that for a pair of concepts (c_i, c_j), we model only (c_i, c_j) as a random variable and not both (c_i, c_j) and (c_j, c_i) as indicated in Definition 1.

Prior Beliefs. Although probabilities $P(r_{ij})$ are not known a priori, we assume that for each r_{ij}, there exists evidence e_{ij}, consisting of features that can be observed (computed) to infer the probability of the concept pair (c_i, c_j) having the relationship R. This allows us to compute a prior for $P(r_{ij} \mid e_{ij})$. We denote the set of observed evidence over all random variables of G as $\mathcal{E}$ and the prior prediction of r_{ij} as f_{ij} where $f_{ij} = \arg\max_{r \in dom(r_{ij})} P(r_{ij} = r)$. We defer the discussion of obtaining these prior beliefs to Section 3.1.

Transitivity: Axiom on Relationships. In addition to the evidence used as priors for relationships, the considered types of relationships in our problem setting, equivalence and parent-child, impose additional structural properties, which can be leveraged for improving relationship predictions. Let us focus on transitivity which introduces dependencies among pairwise relationships. For example, the transitivity condition requires that if an equivalence relationship exists between concept pairs (c_i, c_j) and (c_j, c_k), then it must also exist between (c_i, c_k). Formally, considering equivalence or parent-child, for given predictions f_{ij}, f_{ik}, and f_{jk} evaluated for three concepts c_i, c_j, and c_k, if $f_{ij} = f_{jk} = 1$, transitivity requires that f_{ik} must also be 1. However, if these predictions are made independently, it is possible for f_{ik} to be erroneously predicted, violating the transitivity constraint. The transitivity property of the relationship further implies that certain relationship assignments are inherently inconsistent. Specifically, if we aim to infer f_{ij} and f_{jk} while knowing $f_{ik} = 0$, transitivity dictates that f_{ij} and f_{jk} cannot both be 1, as illustrated in Figure 2. This restriction prevents certain relationship assignments from coexisting.

Conflicting predictions can arise if pairwise decisions are made independently and it requires resolution of inconsistencies to maintain transitivity. For example, flipping either f_{ij} or f_{jk} would resolve the conflict by ensuring that the assignments align with the transitivity constraint. Yet, determining which prediction to alter is a non-trivial task, as it may require additional context. To address these challenges, we introduce a probabilistic formulation and a dependency-aware solution below.

Provided that prior beliefs for each r_{ij}, namely $P(r_{ij} \mid e_{ij})$, are available and we are given some axiom $\mathcal{A}$ on the relationships, the problem of metadata integration can be cast as an optimization problem. More specifically, we consider the common and intuitive transitivity axiom discussed above for our setting. The optimization objective is to find the relationship graph assignment $\tilde{G}$ with the maximum probability, conditioned on observed evidence $\mathcal{E}$, while satisfying the constraints implied by the relationship axiom $\mathcal{A}$.

Definition 2 (Optimal Relationship Graph)**.** Given a relationship R, a set of concepts $\mathcal{V}$, and axiom $\mathcal{A}$, the optimal relationship assignment graph $\tilde{G}$ is the graph with the maximum joint probability of random variables $\{r_{ij}\}_{i<j}$, conditioned on evidence $\mathcal{E}$ and being subject to axiom $\mathcal{A}$.

$$\tilde{G} = \arg\max_{G} P(G \mid \mathcal{E}, \mathcal{A}) = \arg\max_{\{r_{ij}\}_{i<j}} P(\{r_{ij}\}_{i<j} \mid \mathcal{E}, \mathcal{A})$$

This formulation permits a theoretically optimal solution by exploiting a probabilistic graphical model called Markov Random Field, which also allows for incorporating the relationship transitivity and dependencies during the probabilistic inference. We give a brief introduction to Markov Random Field in Section 3.2.

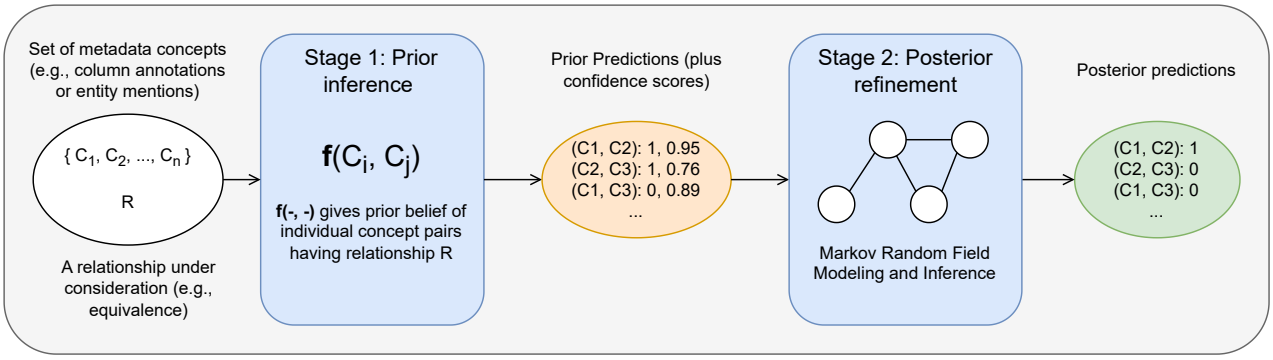

Figure 3: Overview of the proposed two-stage prior-posterior framework for integrating metadata concepts.

3 OPENFORGE FRAMEWORK

As illustrated in Figure 3, we introduce OpenForge, a prior-posterior framework for integrating metadata concepts. Given a set of concepts and a relationship, the framework operates in two stages. In the first stage, local predictions are generated as prior beliefs for each pair of concepts. Various methods may be used to obtain these prior predictions (Section 3.1). In the second stage, we address the relationship transitivity and dependencies between prior predictions through a probabilistic modeling approach using Markov Random Field (Section 3.2). This graphical model encodes the transitivity constraints and leverages labeled data to learn dependencies in a data-driven manner, effectively capturing the underlying relationship structures and refining the prior predictions.

3.1 Obtaining Prior Beliefs

Our probabilistic formulation requires a probability distribution for each random variable. We explore three different methods for obtaining these prior beliefs.

3.1.1 Prompting Large Language Models. Recent advances in large language models (LLMs) have shown human-expert performance on a broad array of natural language processing and multi-modality tasks [2, 15, 44]. These models are pre-trained on a vast amount of curated data and can be applied out-of-the-box to new contexts following a text-to-text paradigm with instruction tuning or prompting [9, 48]. Motivated by such success, we supply an LLM with a task description and prompt the model for predictions and confidence scores of concept pairs. Besides being straightforward, this method requires no labeled data or only few labeled data for demonstration (i.e., few-shot learning), making it feasible for scenarios where labeled data are scarce or there are not enough data for training/fine-tuning a model.

3.1.2 Fine-Tuning Large Language Models. Since LLMs are primarily pre-trained for generation tasks, we also experiment with fine-tuning them for domain-specific classification tasks if training data are available. Due to the large size of LLMs and the hardware constraints, we choose to fine-tune open-source models that have fewer than 10 billions of parameters. In particular, We employ Low-Rank Adaptation (LoRA) [24], a parameter-efficient fine-tuning technique that introduces a small set of trainable parameters (known as adapters) while leaving most of the model's original parameters unchanged. This method enables effective and efficient adaptation of an LLM to a specific relationship classification task with a single GPU. Additionally, LoRA allows for task flexibility: different tasks can share the base LLM, requiring only a switch in adapters rather than maintaining a fully fine-tuned model for each task.

3.1.3 Training Machine Learning Models. As compared to the previous methods based on LLMs, we also manually design task-specific features and train a machine learning model, such as Random Forest, to make predictions. This method serves as a baseline to evaluate the LLMs' effectiveness in capturing dataset semantics.

Note that we fine-tune LLMs or train traditional machine learning models to obtain prior beliefs when training data are available. Nevertheless, all these methods treat each pair of concepts independently and make individual predictions ignoring relationship dependencies. There is no guarantee of avoiding conflicting predictions. Further details on the models and adaptations used in our experiments are provided in Section 5.1.

3.2 MRF Modeling of Metadata Integration

3.2.1 Markov Random Field Primer. Markov Random Field (MRF), also referred to as Markov Network in the literature, is a specific type of undirected graphical model for representing the joint probability distribution of a set of random variables [46].

Specifically, an MRF is defined by an undirected graph $M = (\mathcal{N}, \mathcal{L})$ where $\mathcal{N}$ denotes the set of nodes representing random variables and $\mathcal{L}$ denotes the set of edges representing the statistical dependencies between the random variables.

In MRFs, the joint probability distribution of random variables is factorized as a product of potential functions, each of which is associated with a clique in the graph. The potential function, also known as factor, is a function that assigns a non-negative value to each possible configuration of the random variables in the clique. Here, a configuration of random variables is an assignment of values to random variables. The joint probability of random variables is then proportional to the product of the potential functions over all cliques in the graph, defined as follows.

$$P(x_1, ..., x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(x_c) \quad (1)$$

where x_i denotes a random variable in $\mathcal{N}$, C denotes the set of cliques in G, while ϕ_c denotes a factor defined over a set of variables

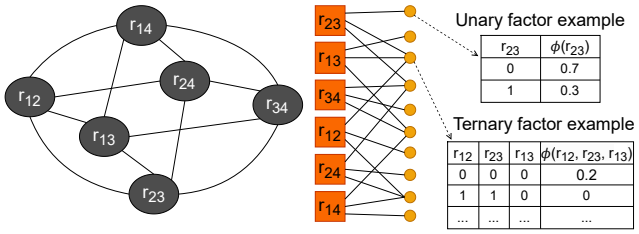

Figure 4: The plot on the left demonstrates an instance of our proposed MRF model containing six nodes/random variables and their dependencies; the plot on the right, known as a factor graph, visualizes the correspondence between factors and random variables in the MRF.

x_c in a clique $c \in C$, and Z is a normalizing constant. Note that clique set C and potential functions $\phi(c)$ need to be specified per application context.

An MRF can also be represented by a factor graph which explicitly shows the correspondence between random variables and factors. Specifically, a factor graph is a bipartite graph where the nodes are divided into two disjoint sets, one for the random variables and the other for the factors. An edge exists between a factor node and a variable node if the factor depends on that variable. This representation simplifies the computation of probability distributions in MRFs, such as Maximum a Posteriori (MAP) inference, which determines the most likely assignment of values to the variables in the MRF. We introduce our MRF design below and discuss MAP inference in Section 4.1.

3.2.2 Our MRF Design. MRFs are powerful models for capturing prior beliefs about and the dependencies between random variables. This makes MRFs a natural fit for our problem formulation, which require modeling existing knowledge about concepts and their relationships including encoding the transitivity axiom.

Let us first describe how the relationship assignment graph, its probability of Definition 1, as well as the transitivity axiom are modeled as a discrete MRF described in Section 3.2.1. We treat the presence of a specified relationship between a pair of concepts (c_i, c_j), an edge between nodes c_i and c_j (directed if applicable) in a relationship graph, as a categorical random variable r_{ij}. We consider two types of potential functions, or factors, in our problem. The first type is unary cliques C_u to capture prior beliefs, each of which contains an individual random variable r_{ij}. The second type is ternary cliques C_t to capture the relationship transitivity and dependencies, each of which consists of three random variables (r_{ij}, r_{jk}, r_{ik}) for $i < j < k$. Note that ternary cliques are the cliques of the smallest size on which we can encode the relationship transitivity. Although encoding the transitivity with higher-order cliques is possible, it will make the model more complex (e.g., the total number of possible configurations of a factor grows exponentially with respect to the size of a clique) and incur additional computational cost (we show that the time complexity of inference over unary and ternary cliques is already quartic with respect to the number of concepts in Section 4.1). Admittedly, the design choice of using ternary cliques instead of higher-order cliques weighs the efficiency over the potential benefit of having a more complex while expressive model.

The left plot of Figure 4 demonstrates an instance of MRF consisting of six nodes/random variables $\{r_{ij} \mid 1 \le i < j \le 4\}$ induced by four concepts $\{c_1, c_2, c_3, c_4\}$. The MRF is densely connected but not fully connected where there is an edge between a pair of nodes if they involve a common concept. For instance, there is an edge between r_{12} and r_{13} because they both involve concept c_1 whereas there is no edge between r_{12} and r_{34}. Under the hood, the edge connectivity is attributed to ternary cliques we consider in the MRF. We do not include all ternary cliques of three random variables in the MRF (e.g., the clique of r_{12}, r_{13} and r_{34}) but ternary cliques that involve exactly three concepts for the purpose of modeling the relationship transitivity. The right plot of Figure 4 shows the factor graph representation of the MRF that visualizes the correspondence between factors and random variables. This factor graph consists of six random variables and ten factors. Each variable is associated with three factors while each unary factor is associated with one random variable and each ternary factor is associated with a clique of three variables.

Since an MRF computes the joint probability of the graph based on cliques in the graph, now we can decompose the probability of our relationship assignment graph following Equation 1 as below.

$$P(G \mid \mathcal{E}, \mathcal{A}) = P(\{r_{ij}\}_{i<j} \mid \mathcal{E}, \mathcal{A}) = \frac{1}{Z} \prod_{c \in C} \phi_c(\{r_{ij}\}_{i<j, r_{ij} \in c} \mid \mathcal{E}, \mathcal{A})$$
$$= \frac{1}{Z} \prod_{c \in C_u} \phi_c(r_{ij} \mid \mathcal{E}) \cdot \prod_{c \in C_t} \phi_c(r_{ij}, r_{jk}, r_{ik} \mid \mathcal{A}) \quad (2)$$

The potential function for each unary clique $\phi_c(r_{ij} \mid \mathcal{E})$ is defined as the conditional probability of the relationship's presence given the observed evidence of the concept pair, allowing us to capture the prior belief of each individual random variable.

While each unary clique has its own potential function, all ternary cliques share a common potential function, which is parameterized to capture relationship dependencies. Specifically, configurations that violate transitivity (i.e., invalid configurations) are assigned a potential value of zero, while each valid configuration is parameterized to represent its likelihood. Higher parameter values indicate a greater probability for the corresponding assignment within the ternary clique. Table 1 gives an example of the parameterized potential function for ternary cliques of binary random variables, where 0 represents no relationship and 1 represents an equivalence relationship. Among the eight possible configurations of three binary variables, three configurations (i.e., [0, 1, 1], [1, 0, 1], and [1, 1, 0]) are invalid and thus set to zero. The remaining configurations are parameterized to determine their potential function values. Although it is technically feasible to assign unique potential functions to each ternary clique, this would result in a model with a linear increase in parameters relative to the number of ternary factors, making the model more complex. To maintain simplicity and prevent over-parametrization, we instead use a shared potential function across all ternary cliques. Hence, Equation 2 can be written as follows.

$$P(G \mid \mathcal{E}, \mathcal{A}) = \frac{1}{Z} \prod_{r_{ij} \in G} P(r_{ij} \mid e_{ij}) \cdot \prod_{c \in C_t} \phi(r_{ij}, r_{jk}, r_{ik} \mid \mathcal{A}) \quad (3)$$

Table 1: Example of the parameterized potential function ϕ for ternary cliques of binary random variables r_{ij}, r_{jk} and r_{ik} where 0 represents no relationship and 1 represents an equivalence relationship. The potential values $\{\theta_i\}_{i=1}^{5}$ are learnable parameters.

r_{ij}	0	0	0	0	1	1	1	1
r_{jk}	0	0	1	1	0	0	1	1
r_{ik}	0	1	0	1	0	1	0	1
$\phi(r_{ij}, r_{jk}, r_{ik})$	θ_1	θ_2	θ_3	0	θ_4	0	0	θ_5

where probabilities $P(r_{ij} \mid e_{ij})$ are given by some prior model and ϕ denotes the shared parameterized potential function for ternary cliques, which is independent of prior beliefs.

The parametrization of the shared potential function for ternary cliques enables us to learn relationship dependencies such as the label distributions in the dataset. In practice, datasets often exhibit skewed class proportions. For example, in an entity matching dataset, there are typically far more non-equivalent pairs of entities than equivalent ones. This imbalance affects the configuration of random variables in ternary cliques. In the parameterized potential function example in Table 1, the configuration [0, 0, 0] is expected to have a higher potential value than other configurations, reflecting the larger number of concept pairs with no relationship. To learn the parameters in the shared potential function, we treat it as a hyperparameter tuning problem. Rather than manually setting these parameters, we search for well-performing parameters on a labeled validation set using a Bayesian Optimization technique [35]. We discuss automatic parameter tuning in Section 5.2.

Provided the learned potential function for ternary cliques, performing Maximum a Posteriori inference over this MRF gives us posterior predictions, an assignment to random variables yielding the maximum joint probability. Algorithm 1 summarizes our MRF modeling approach. Line 1-2 initialize the MRF with random variables. Lines 3-7 utilize methods, from Section 3.1, to estimate the conditional probabilities of random variables, and add unary factors to the MRF. In lines 8-14, we enumerate ternary cliques to create the ternary factors and line 15 executes the inference algorithm.

We remark that for a set of n concepts, the MRF we propose consists of $\binom{n}{2} = \frac{n(n-1)}{2}$ nodes and $O(n^3)$ factors (more precisely, $\binom{n}{3}$ factors for ternary cliques plus $\binom{n}{2}$ factors for unary cliques). Section 4.1 describes MRF inference and presents an analysis on the time and space complexity of running inference on our proposed MRF model. Section 4.2 discusses how we scale our MRF modeling for large and sparse datasets.

Algorithm 1: MRF Modeling

Input : V, a set of concepts; f, a model that gives prior beliefs; ϕ, a shared potential function for ternary cliques; g, a MRF inference algorithm

Output : G, a relationship assignment graph

```
1  mrf ← init_mrf( ) ;
   /* Create random variables for each ordered pair
      of concepts from the initial vocabulary     */
2  rvs ← V ;
   /* Add nodes and factors for unary cliques     */
3  foreach var ∈ rvs do
4  |  mrf.add_node(var), prior_prob ← f(var) ;
5  |  unary_factor ← init_factor([var], prior_prob) ;
6  |  mrf.add_factor(unary_factor) ;
7  end
   /* Add edges and factors for ternary cliques   */
8  ternary_cliques ← generate_ternary_cliques(rvs) ;
9  foreach (var1, var2, var3) ∈ ternary_cliques do
10 |  mrf.add_edges_from(
11 |      [(var1, var2), (var2, var3), (var1, var3)]) ;
12 |  ternary_factor ← init_factor([var1, var2, var3], ϕ) ;
13 |  mrf.add_factor(ternary_factor) ;
14 end
15 G ← g(mrf); /* Run MRF inference algorithm     */
16 return G
```

4 SCALABLE METADATA INTEGRATION

4.1 Inference in Markov Random Field

In our problem, we are interested in finding the most probable relationship graph or equivalently, the most probable assignment to random variables given the observed evidence, i.e., Maximum a Posteriori (MAP) inference. Performing exact MAP inference on complex non-tree structured MRFs is known to be NP-hard and computationally intractable when the graph is large and contains many loops [17, 28]. For example, a dataset containing 1000 concepts results in half a million of random variables and over 166 millions of factors. We thus resort to approximate inference algorithms, which mainly fall into three categories: linear programming-based inference, sampling-based inference, and variational inference [17].

4.1.1 Approximate MAP inference. To find the most efficient existing inference algorithms, we evaluated various implementations from each category of approximate inference algorithms (more details in Section 5.2). We found that Loopy Belief Propagation (LBP) [28, 40], a special case of variational inference algorithms, is most efficient and scalable for our MRF model design. Our finding is consistent with the literature where variational inference methods often scale better and are more amenable to optimizations such as parallelization over multiple processors and hardware acceleration using GPUs [17, 64].

LBP is a message-passing algorithm on a loopy graph that iteratively updates the belief of each random variable based on the beliefs of its neighbors in the graph. More precisely, we use the max-product LBP algorithm on factor graphs to perform MAP inference and handle high-order potentials (i.e., potentials of ternary cliques). With respect to the factor graph, LBP computes the message $m_{x\to w}(x_i)$ from a variable node x to a factor node w (here factor w contains variable x so there is an edge between the two nodes for message passing) as $m_{x\to w}(x_i) = \prod_{y\in Nb(x)\setminus w} m_{y\to x}(x_i)$, where $Nb(x)$ denotes the set of neighbors of node x, and x_i is a value that x can take. Conversely, the message $m_{w\to x}(x_i)$ from

factor node w to variable node x is computed as $m_{w\to x}(x_i) = \max_{I, I_x = x_i}(\phi(I) \prod_{y \in Nb(w)\setminus x} m_{y\to w}(y_i))$, where I is a valid assignment to all variables in factor w, tuple $(I, I_x = x_i)$ denotes a valid assignment with $x = x_i$, and ϕ is the potential function corresponding to w. Once a variable x receives messages from all its neighbors in an iteration, the maximal belief of x can be computed as $x^* = \arg\max_{x_i} \prod_{y \in Nb(x)} m_{y\to x}(x_i)$.

4.1.2 Time Complexity Analysis. Having described the MRF approximate inference algorithm we apply for finding the optimal relationship assignment graph, let us describe its complexity. Let s and d be the number of possible states and degree of variable x, respectively. Computing a variable-to-factor message $m_{x\to w}(x_i)$ requires combining all incoming messages from neighboring factor nodes excluding w for each state of x. Then the cost of computing a single variable-to-factor message is $O(s \cdot (d-1))$. It follows that the time complexity for computing all variable-to-factor messages in one iteration is $O(\sum_x d \cdot s \cdot (d-1)) = O(|V| \cdot s \cdot d^2)$ where $|V|$ denotes the number of variable nodes in the factor graph.

Let d_w be the degree of factor node w. Computing a factor-to-variable message $m_{w\to x}(x_i)$ requires maximizing over the values of all variables connected to w excluding x. Then, the cost of computing a single factor-to-variable message is $O(s^{d_w - 1})$ as there are s^{d_w-1} configurations of connected variables excluding x. It follows that the time complexity for computing all factor-to-variable messages in one iteration is $O(\sum_w d_w \cdot s^{d_w})$. Combining the complexities for both types of messages, the time complexity of running max-product LBP for l iterations is $O(l \cdot (|V| \cdot s \cdot d^2 + \sum_w d_w \cdot s^{d_w}))$.

In our proposed MRF model, there are $\binom{n}{2}$ binary variable nodes with degree $n-1$, $\binom{n}{2}$ unary factors with degree 1 and $\binom{n}{3}$ ternary factors with degree 3 given a set of n concepts. By plugging in these numbers, the time complexity above can be simplified to $O(l \cdot n^4)$.

4.1.3 Space Complexity Analysis. Recall from Figure 4 that factors are modeled as bipartite graphs between variables and factors. Each edge in the factor graph requires two messages—one message from the variable node to the factor node and another from the factor node to the variable node. Storing a message requires space proportional to the number of states of a variable. Then, the space complexity of storing messages is $O(|E| \cdot s)$, where $|E|$ denotes the number of edges in the factor graph. Additionally, for each variable node, we store a belief over its possible states, which requires $O(s)$ space per variable. For $|V|$ variable nodes, the space required for storing beliefs is $O(|V| \cdot s)$. Thus, the overall space complexity for max-product LBP is $O(|E| \cdot s + |V| \cdot s)$ which translates to $O(n^3)$ in our proposed MRF model.

The quartic time complexity and cubic space complexity is forbidding for large datasets with thousands of concepts. Next, we introduce a solution to scale up the MRF inference.

4.2 Scaling MRF Modeling

The high time and space complexity stems from the densely connected graph we construct. For a set of n concepts, each node in the resulting factor graph connects to $n-1$ neighbors including one unary and $n-2$ ternary factors. This dense connectivity leads to significant computational demands and memory consumption.

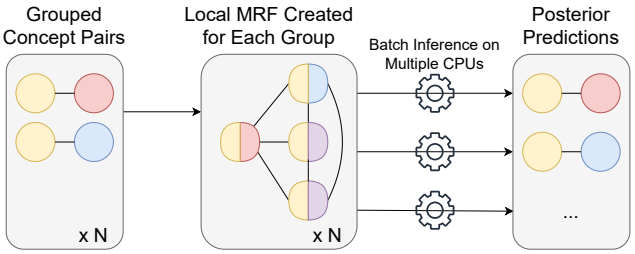

Figure 5: Creating independent MRFs for concept pairs in large datasets with sparse relationships. Ordered pairs are first grouped by the left concept and top-k neighbors (represented by the purple semi-circles) are retrieved for the left concept to construct a local MRF of random variables. Inference over independent MRFs is parallelized on available CPUs and posterior predictions of test pairs are collected.

However, we empirically observe that many connections in the dense graph are unnecessary for ensuring relationship transitivity and learning dependencies. Specifically, numerous connections are either redundant or represent weak dependencies that do not significantly impact inference accuracy. We propose to systematically reduce the connections for each node. By reducing the connections from $n-1$ to a constant k, where $k << n$, we effectively decrease the number of message-passing operations required by the LBP algorithm, enabling faster execution and significantly lower memory usage. Our intuition is that concepts with more similar embeddings are more likely to share strong relationship dependencies. Therefore, to select these k neighbors for each concept, we employ a text embedding model, `NV-Embed-v2` [32] to represent concepts in embedding space and search for k-nearest neighbors based on Cosine similarity. By focusing only on the top-k neighbors, we maintain the essential relationship dependencies while disregarding redundant connections that contribute little to inference quality.

Reducing the number of connections for each node has the advantage of dividing the MRF into disconnected local graphs. Our solution first groups ordered concept pairs by the left concept and constructs a local MRF for each group. This ensures that the same concept pair does not appear in multiple MRFs and local MRFs are independent of each other. The inference for each local MRF can then be performed independently in constant time. Furthermore, we can parallelize the inference for local MRFs across the available CPUs, which significantly reduce the inference time. Figure 5 provides an illustration of the idea.

5 IMPLEMENTATION

5.1 Prior Models

5.1.1 Large Language Models. We employ two open-source LLMs as prior models for prompting and fine-tuning: `gemma-2-9b-it` from Google [36] and `Qwen2.5-7B-Instruct` from Alibaba [61]. Both models can be accessed from Hugging Face's model hub. We choose these two models as we are able to deploy them locally and they can closely follow our straightforward prompts to generate both predictions and confidence scores. We refer to both models as gemma-2 and qwen2.5 respectively for convenience below.

Due to the space limitation, we have made our prompts and fine-tuning hyperparameters publicly available in our GitHub repository. Both models can be fine-tuned efficiently under an hour (for 20 epochs). One noteworthy issue we encountered when fine-tuning gemma-2 and qwen2.5 is that both models tend to exhibit excessive confidence in their predictions, with most predictions yielding a probability score above 0.99. This overconfidence hinders the message-passing process in MRF inference, as only information from the dominant class gets propagated. To mitigate the issue, we incorporate three techniques for calibrating model confidence. Specifically, we use a weighted loss function (with weights inversely proportional to the frequency of the respective classes) when fine-tuning models on imbalanced datasets and enable label smoothing [39] in the loss calculation, which assigns a small probability to incorrect classes to reduce overfitting. Additionally, we apply temperature scaling [21] to adjust the sharpness of the output logits during inference, which involves dividing the logits by a temperature value greater than 1 before applying the softmax function, thereby softening the resulting probabilities.

5.1.2 Machine Learning Models. We train multi-class classifiers on a predefined set of features to predict the presence of a relationship for each concept pair. We employ three classifiers including Ridge classifier, Random Forest, and Gradient-Boosted Decision Tree from scikit-learn [47]. These models were chosen for their built-in regularization mechanisms, which help reduce overfitting and improve generalization on class imbalanced datasets. We define features using concept names and associated data values if available. These features include string similarities (Q-gram, Jaccard, and edit distance), embedding similarities (fastText [6]), word and character count ratios between concept names, and Jaccard and embedding similarities between column values.

5.2 MRF Inference and Parameter Tuning

We evaluated five implementations of MAP inference for MRFs from three libraries: Shafer-Shenoy [51] from `pyAgrum` [16], Belief Propagation [46], Max-Product Linear Programming (MPLP) [19], and Gibbs Sampling [18] from `pgmpy` [3], and Loopy Belief Propagation (LBP) [40] from `PGMax` [64]. LBP from `PGMax` proved the most efficient and scalable due to their flat array-based implementation which leverages just-in-time compilation and parallelization for optimized execution on multi-processors and GPUs.

There are two sources of parameters we tune for our problem: those in the potential function of ternary cliques and those for LBP inference (e.g., the damping factor used to improve the convergence of LBP). In particular, we tune the parameters using SMAC [35], a robust and flexible Bayesian Optimization framework for hyperparameter optimization. SMAC treats MRF inference as a black-box function and iteratively searches for parameter configurations that maximize a target function which gives accuracy or F1 score for example. Internally, SMAC employs a surrogate model to approximate the target function and in each iteration proposes a configuration with the maximum expected improvement (which is computed using the surrogate model) for evaluation. The optimization process balances between exploration (i.e., search an unknown region in the parameter space) and exploitation (i.e., search for configurations near the best-so-far configuration), and stops after a fixed number of iterations. SMAC is known to be effective and efficient for AutoML applications [35] and under active maintenance.

Table 2: Summary of dataset statistics in the training, validation and test splits. All four datasets exhibit class imbalance.

Datasets	SOTAB	Walmart-Amazon	ICPSR-Detection	ICPSR-Direction
Relationship Type	Equivalence	Equivalence	Parent-Child	Parent-Child
Number of Concepts	46 / - / 46	5126 / 2507 / 2484	140 / 54 / 54	79 / 46 / 46
Number of Pairs	1035 / - / 1035	6144 / 2049 / 2049	9730 / 1431 / 1431	3081 / 1035 / 1035
% of Minority Class	1.26% / - / 1.26%	9.38% / 9.42% / 9.42%	1.63% / 4.05% / 3.98%	2.08% / 3.29% / 3.29%

6 EXPERIMENTS

6.1 Experimental Setup

6.1.1 Datasets. We evaluate OpenForge on four datasets involving two relationship types: equivalence (SOTAB dataset for column types matching and Walmart-Amazon dataset for entity matching) and parent-child (ICPSR-Detection and ICPSR-Direction datasets for taxonomy induction). We have sourced SOTAB and ICPSR datasets particularly for metadata integration tasks.

SOTAB Dataset. The Schema.org Table Annotation Benchmark (SOTAB) [29, 30] provides manually verified column type annotation using two vocabularies, `schema.org` [20] and DBPedia [33]. We merge concepts from both vocabularies to form an initial collection of concepts. If a table column is annotated with concepts from both vocabularies, we consider the two concepts equivalent. The many-to-many correspondences from columns to concepts can challenge the priors. This allows us to test how the proposed MRF formulation can identify conflicting relationships. Given the limited vocabulary size, we split it evenly into training and test sets.

Walmart-Amazon Dataset. This dataset has been widely used to evaluate entity matching algorithms [38, 58]. It provides a challenging setting (e.g., noisy attributes and missing data) for matching product entities between two major e-commerce platforms. We use the same training, validation, and test splits as prior work.

ICPSR-Detection and ICPSR-Direction Datasets. ICPSR, the institute maintaining the world's largest social science data archive [26], makes their controlled vocabularies publicly available. We create two datasets of parent-child relationships from ICPSR's Subject Thesaurus [25]. Following the task definitions in [49], the ICPSR-Detection dataset is for hypernymy detection where one predicts whether the hypernymy relationship exists between a pair of concepts (i.e., binary classification) while for the ICPSR-Direction dataset, one needs to identify which concept is broader in a given pair of concepts if the relationship exists (i.e., multi-class classification). The dataset comprises social science terms from various disciplines such as political science and economics. We manually identify concepts with transitive parent-child relationships (e.g., `credentials` → `academic degrees` → `doctoral degrees`) and divide the collection of transitive concept tuples into training, validation, and test sets at ratio of 0.6:0.2:0.2. The ICPSR datasets have no column data associated with each concept as ICPSR does not use subject terms to annotate table elements. The prior beliefs of these two datasets can only be learned from concept names, which is a more challenging setting.

Table 2 summarizes dataset statistics. Notably, all datasets exhibit highly imbalanced class distributions, with minority class proportions in test splits ranging from 1.26% to 9.42%, which poses a significant challenge for any solution.

6.1.2 Baselines. For each task defined by a relationship, we compare our approach with the previous state-of-the-art (SOTA) methods as well as latest LLMs, GPT-3.5 and GPT-4, from OpenAI.

Unicorn. Unicorn [58] is a unified multi-task model for data integration tasks like entity and schema matching. It leverages multi-task learning and a Mixture-of-Experts architecture to enable knowledge sharing across tasks and datasets. Additionally, Unicorn claims to support zero-shot prediction for new tasks without requiring labeled data. We compare OpenForge with Unicorn on datasets involving equivalence relationships.

COMA. COMA [14] combines multiple match algorithms to improve the schema matching accuracy. Later extensions of COMA offer ontology matching among other features.

ADA. Zhang et al. [63] propose a data-driven solution for automatically discovering attributes in relational databases, which we refer to as ADA for convenience. ADA clusters columns into attributes based on data distribution similarities where each attribute represents a unique semantic concept (e.g., telephone numbers).

Ditto. Ditto [34] casts entity matching as a sequence-pair classification task and fine-tunes pre-trained language models with labeled data and optimizations to achieve strong performance.

COMA and ADA, recognized as strong schema matching baselines [31], are compared with OpenForge on the SOTAB dataset, where the setup is most aligned with schema matching. We include Ditto for comparison on the Walmart-Amazon dataset as it is the task-specific SOTA method for entity matching.

Chain-of-Layer. Zeng et al. [62] propose Chain-of-Layer for automatic taxonomy induction from a given set of entities. It heavily relies on prompting a LLM to construct a taxonomy from top to bottom in an iterative manner and uses another smaller language model to remove hallucinated entities and relationships.

HypernymySuite. Roller et al. [49] investigates hypernymy detection through two primary approaches: pattern-based methods, which utilize predefined lexico-syntactic patterns to identify hierarchical relationships between words, and distributional methods, which leverage word co-occurrence statistics in large corpora.

We compare OpenForge with Chain-of-Layer and HypernymySuite on datasets involving parent-child relationships.

GPT-3.5/4. We prompt GPT-3.5 and GPT-4 [9, 44] to predict relationships between concept pairs using task descriptions and few-shot learning. We use OpenAI APIs to interact with the models identified as `gpt-3.5-turbo-0125` and `gpt-4-turbo-2024-04-09`.

Due to class imbalance in the datasets, we evaluate the quality of all approaches using F1 score (macro F1 for multi-class classification) and report the best results of each approach. For example, LLMs can obtain higher scores without few-shot learning in some cases. Additionally, we report the runtime of our MRF modeling in seconds for efficiency and scalability analysis.

6.1.3 Hardware. Experiments involving GPU-accelerated MRF inference, fine-tuning and inference of open-source LLMs, and runtime measurement are done on a node with a NVIDIA A40 40GB GPU, 16 cores of Intel Xeon Gold 6226R processors and 256 GB of RAM on a shared computing cluster. The rest of experiments are conducted on a local server with 16 cores of Intel Xeon Bronze 3106 processors and 256 GB of RAM.

Table 3: Comparison of F1 score across methods and datasets.

Datasets	Methods					
	Unicorn	COMA	ADA	GPT-3.5	GPT-4	OpenForge
SOTAB	0.48	0.59	0.10	0.73	0.75	**1.00**
	Unicorn	Ditto		GPT-3.5	GPT-4	OpenForge
Walmart-Amazon	0.87	0.87		0.67	0.84	**0.91**
	Chain-of-Layer	HypernymySuite		GPT-3.5	GPT-4	OpenForge
ICPSR-Detection	0.37	0.21		0.58	0.79	**0.91**
ICPSR-Direction	0.01	0.34		0.41	0.45	**0.81**

6.2 Quality Comparison

RQ1: How does OpenForge compare with baselines for various metadata integration tasks? As shown in Table 3, OpenForge consistently outperforms the baselines by significant margins in F1 score across all datasets (e.g., 4-36 F1 points improvement over the second-best baselines). These results highlight OpenForge's flexibility and robustness in capturing different relationship types, outperforming both specialized and general-purpose baselines. We attribute the superior performance to three key factors:

(1) Joint Probability Modeling: MRF effectively models the joint probability over random variables, especially with strong priors given by fine-tuned LLMs, enabling more accurate predictions.
(2) Preservation of Transitivity: By incorporating potential functions for ternary cliques, OpenForge ensures that invalid assignments violating relationship transitivity are avoided.
(3) Dependency Learning: Fine-tuning hyperparameters of MRF modeling and inference on a validation dataset with a class distribution similar to the test set allows OpenForge to capture relationship dependencies more effectively.

We discuss some highlighted results on each dataset below.

6.2.1 SOTAB dataset. OpenForge achieves a perfect F1 score of 1.0 even though the prior beliefs used as inputs to MRF modeling are far from perfect. We report a detailed comparison of prior models and OpenForge (prior beliefs + MRF-based modeling) in Section 5.1. These results provide strong evidence for the advantage of modeling relationship transitivity and dependencies using MRFs. Among the baselines, GPT-4 is the second-most effective method but trails OpenForge by 25 F1 points. COMA employing multiple matching strategies exhibits decent F1 score of 0.59 while another traditional method ADA using only the Earth Mover's Distance as the similarity measure appears to be the least effective. Unicorn, the SOTA for data matching tasks, performs less competitively on this dataset. The suboptimal performance suggests that Unicorn, despite leveraging multi-task learning, finds it challenging to generalize to the metadata integration tasks and datasets. This is most likely due to two reasons: (1) the SOTAB dataset is not seen in pretraining of Unicorn, and (2) equivalent pairs in Unicorn datasets often share common parts whereas column values associated with equivalent types in SOTAB are heterogeneous and may have no overlap.

6.2.2 Walmart-Amazon dataset. OpenForge achieves an F1 score of 0.91, outperforming Unicorn, the second-best method, by 4 F1 points. The performance gap is smaller on this dataset as Unicorn was pre-trained on multiple entity matching datasets including the Walmart-Amazon dataset, making it a strong baseline. Additionally, the sparsity characteristic of this dataset (i.e., many entities appear only once in the test set and the predictions of many entity pairs are independent of others) diminishes the advantage of explicitly modeling relationship transitivity and dependencies. Ditto, the SOTA task-specific approach for entity matching, performs comparably to Unicorn due to its use of supervised learning. Among other baselines, GPT-4 falls behind OpenForge by 7 F1 points, while GPT-3.5 lags even further, trailing by 24 F1 points.

6.2.3 ICPSR-Detection and ICPSR-Direction datasets. OpenForge outperforms GPT-4 by 12 F1 points and GPT-3.5 by 33 F1 points on the ICPSR-Detection dataset, and by 36 F1 points and 40 F1 points, respectively, on ICPSR-Direction. The specialized approach, Chain-of-Layer, performs poorly in both tasks due to its assumption that all concepts belong to the same domain and fit within a single structured hierarchy, which does not hold for these datasets. The pattern-based approach, HypernymySuite, performs the worst on ICPSR-Detection, as many concepts in the dataset are absent from the corpus it relies on for pattern extraction and it has limited capability in handling out-of-vocabulary concepts. While HypernymySuite obtains a moderate score on ICPSR-Direction, it still falls behind due to its reliance on extracted patterns, which struggle with directionality inference, particularly for unseen concepts.

6.3 Prior Models

RQ2: Can MRF modeling improve on various prior beliefs and by how much? Figure 6 illustrates the impact of prior models on the result quality of OpenForge for the SOTAB, Walmart-Amazon, and ICPSR datasets. To save space, we present results for up to five prior models: the Ridge classifier (the top-performing ML model among three tested), gemma-2, qwen2.5, and their LoRA fine-tuned classifier variants, i.e., gemma-2-lora and qwen2.5-lora. On all datasets, our MRF modeling significantly improves the results quality over the prior models, with gains ranging from 2 to 70 F1 points. These results demonstrate the effectiveness of MRF modeling in enhancing prior predictions and correcting mispredictions that violate relationship transitivity.

We observe that overall LoRA fine-tuned LLMs outperform vanilla LLMs with few-shot learning by a large margin. For instance, the LoRA fine-tuned gemma-2 model achieves an F1 improvement of 52 points over the vanilla gemma-2 model, while the corresponding improvement on the Walmart-Amazon dataset is 45 F1 points. This highlights the potential of fine-tuning vanilla LLMs to obtain strong performance for classification tasks. Notably, on the Walmart-Amazon dataset, MRF modeling does not provide additional improvements for the gemma-2-lora and qwen2.5-lora models. This can be attributed to two factors: (1) the dataset's sparsity, where many entity pairs in the test set are independent, reducing the occurrence of relationship transitivity violations; and (2) fine-tuned LLMs serving as strong priors, with manual checks confirming that none of their mispredictions violate relationship transitivity. Nevertheless, applying MRF modeling does not degrade the quality of the results, preserving the performance of these strong prior models. On the ICPSR-Direction dataset, the regularization effect of MRF modeling is diminished. This is because the ternary factor table has 27 configurations, only four of which violate transitivity, whereas in the binary classification setting, two out of eight configurations are invalid.

Guide for Practitioners. When sufficient training data (e.g., thousands of examples) are available, fine-tuning LLMs using parameter efficient fine-tuning techniques for domain-specific tasks yields very strong results and is highly recommended. In cases where training data are lacking, prompting a LLM to obtain prior beliefs offers a viable alternative.

6.4 Efficiency of MRF Modeling

RQ3: How efficient are MRF construction and inference? Figure 7 shows the MRF construction and inference time of different inference algorithms (i.e., pgmpy-MPLP, PGMax-CPU-LBP, and PGMax-GPU-LBP) on four datasets. All runtime numbers are reported using either a single CPU or GPU. We exclude the Shafer-Shenoy algorithm (implemented in pyAgrum) and Gibbs sampling (implemented in pgmpy) from the comparison as they do not complete execution within an hour on our smallest dataset.

The three algorithms under consideration exhibit efficient factor graph construction, with completion time of a few seconds across all datasets. The primary distinction between these algorithms lie in their inference time. GPU-accelerated LBP (i.e., PGMax-GPU-LBP) is overall the most efficient, achieving more than 2x speedup compared to CPU-based LBP (i.e., PGMax-CPU-LBP) and over two orders of magnitude speedup compared to pgmpy-MPLP on CPU. The speedup is particularly significant on ICPSR datasets, which have over a thousand of random variables and more than twenty thousands of factors in the constructed graph. On ICPSR-Detection, pgmpy-MPLP does not complete running within two and a half hours. In contrast, PGMax-GPU-LBP finishes the inference in just 33.4 seconds and PGMax-CPU-LBP completes in 72.1 seconds. The superior performance of PGMax-GPU-LBP can be attributed to their flat array-based implementation of LBP in JAX [7] which leverages just-in-time compilation and parallelized array operations to optimize performance on hardware accelerators. However, for smaller graphs, such as the local MRFs in the Walmart-Amazon dataset, each of which has tens of random variables due to the dataset's sparsity, pgmpy-MPLP appears to be more efficient while PGMax-GPU-LBP remains 1.3x faster than PGMax-CPU-LBP.

Although PGMax-LBP inference is slower on a single CPU compared to GPU, we can batch process independent local MRFs in the Walmart-Amazon dataset across multiple CPUs. When we use all 16 CPUs on the computing node, the runtime of PGMax-CPU-LBP can be reduced from nearly 1000 seconds to under 70 seconds.

6.5 Scalability

RQ4: How does MRF inference scale as MRF grows in size with a fixed amount of memory? Given that MRF inference is often the most time- and memory-intensive component of the solution, we evaluate its scalability under fixed resources: a single GPU (NVIDIA A40, 40 GB) and 256 GB of RAM. We synthesize

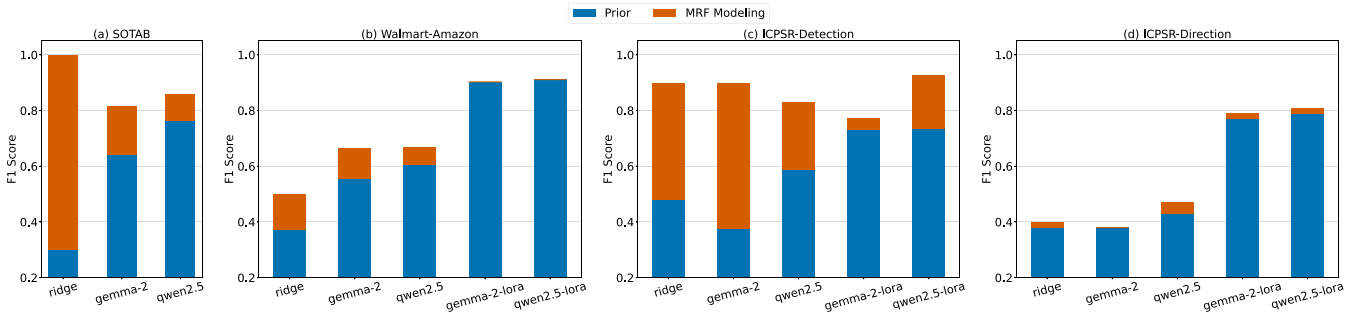

Figure 6: Comparison of F1 score between prior models and OpenForge on (a) SOTAB, (b) Walmart-Amazon, (c) ICPSR-Detection and (d) ICPSR-Direction datasets. We omit gemma-2-lora and qwen2.5-lora on SOTAB due to insufficient training data.

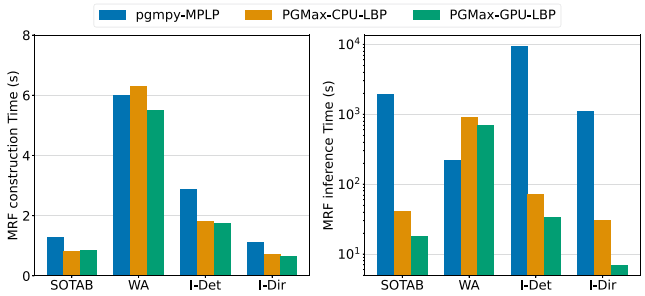

Figure 7: MRF construction and inference time of three inference algorithms using a single CPU or GPU. WA, I-Det, and I-Dir are shorthands for Walmart-Amazon, ICPSR-Detection, and ICPSR-Direction datasets, respectively.

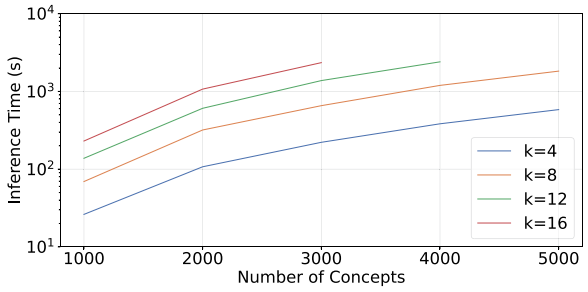

Figure 8: Scalability of MRF inference w.r.t. the number of concepts and connectivity k. $k = 12$ scales up to 4000 concepts (nearly 8 millions of random variables) and $k = 16$ scales up to 3000 concepts (about 4.5 millions of random variables).

sparse factor graphs with varying numbers of concepts (up to 5,000, translating to over 12 million random variables) and connectivity levels k (up to 16) as discussed in Section 4.2. These factor graphs consist of binary random variables with fixed prior probabilities and ternary factors with predefined potential values. We execute the PGMax-GPU-LBP algorithm using its default hyperparameters for 200 iterations.

Figure 8 illustrates the relationship between inference time and the number of concepts for varying connectivity levels. As expected, the inference time grows with both the number of concepts and the connectivity k. The inference times for all completed runs remain within 40 minutes. For connectivity levels $k = 4$ and $k = 8$, the algorithm completes inference on graphs with 5,000 concepts in under 30 minutes. At higher connectivity levels ($k = 12$ and $k = 16$), the algorithm can handle graphs with up to 4,000 and 3,000 concepts, respectively, before exceeding the memory limits. Notably, a graph with 3,000 concepts corresponds to approximately 4.5 million pairs of concepts or random variables for prediction—significantly larger than the test set size of any existing public dataset for data matching or taxonomy induction. Hence, we consider the inference algorithm good enough for handling many use cases.

7 DISCUSSION

More Types of Relationships. While OpenForge currently focuses on equivalence and parent-child relationships, its probabilistic framework can be extended to handle more diverse relationship types. For instance, directional relationships including part-whole and causal dependencies can be modeled similarly to parent-child relationships, with adjustments to the configurations of ternary factors to capture their unique properties.

Beyond Transitivity. While OpenForge primarily enforces transitivity in relationship inference, we acknowledge that some semantic correlations between concept pairs exist independently of transitivity. In Section 4.2, we leverage embedding similarities to identify semantically related concept pairs when constructing local MRFs, indirectly capturing such dependencies. Nevertheless, these semantic correlations are not explicitly modeled within the MRF model, which could further refine relationship assignments.

Future work could extend the model by incorporating pairwise semantic factors directly into the MRF to formally represent such dependencies. This would enable OpenForge to capture richer relational patterns beyond transitivity, further enhancing its expressiveness and robustness in complex metadata integration tasks. Meanwhile, it may introduce additional computational complexity, particularly for large datasets. Balancing model expressiveness with inference efficiency will be a key challenge to tackle in future work.

8 RELATED WORK

We discuss data matching, from the data management community, and taxonomy induction, from the semantic web community, which

are the most relevant to our work. Moreover, we review applications of probabilistic models in the broader area of data integration and summarize the role of metadata in dataset discovery.

Data Matching. Research efforts in generalizing matching techniques have primarily focused on solutions that generalize across domains [53] or tasks [58]. To the best of our knowledge, Unicorn, a multi-tasking model for data matching [58], represents the SOTA in generalization across tasks. It supports matching tasks for data integration, such as entity matching and schema matching, by primarily focusing on equivalence relationships. For (meta)data integration, OpenForge not only goes beyond equivalence relationships to discover and resolve parent-child relationships, but also our experiments show that, despite Unicorn adopting a multi-task learning approach, it is still challenging for Unicorn to generalize to the new matching tasks and datasets of metadata integration. Parallel to multi-task learning, we demonstrate the promising potential of transfer learning using large language models with fewer than 10 billion parameters for various tasks. With parameter-efficient fine-tuning techniques such as LoRA [24], multiple tasks can share the same base model by incorporating lightweight adapters. These adapters are significantly smaller than the base model in size yet effectively enhance result quality across downstream tasks, making this approach highly efficient and scalable.

Taxonomy Induction. The taxonomy induction literature has extensively studied the problem of extracting parent-child relationships from text documents [23, 49, 52]. Various traditional frameworks have posed the taxonomy induction problem with a probabilistic formulation [4, 54]. For example, Snow et al. define the objective as finding a parent-child graph that maximizes the probability of observed evidence [54]. Their formulation relies on two additional independence assumptions to decompose the joint probability so that they can solve the problem with a heuristic search algorithm. This inspired us to design a simpler and more intuitive joint probability formulation, i.e., finding the most probable relationship graph given the existing evidence. This alternative formulation has two advantages. First, it allows us to incorporate various powerful priors, such as LLMs. Second, it allows us to cast our problem to MRF to incorporate relationship properties, such as transitivity, to refine the results. Bansal et al. also propose an MRF formulation for taxonomy induction [4]. OpenForge not only extends this formulation with an additional relationship type, it also explores different ways of obtaining prior beliefs and learning parameters for the MRF formulation. Chain-of-Layer [62], SOTA in the literature, iteratively prompts a LLM to construct the taxonomy while employing a smaller language model to reduce hallucinations of the LLM. However, our experiments reveal that Chain-of-Layer struggles to construct taxonomies effectively when working with a diverse set of metadata concepts from the social science domain. The deficiencies are mainly attributed to the prompts tailored for taxonomies with simple structures (e.g., country-state hierarchies) and the smaller language model's inability to generalize effectively to entities within the social science domain.

Probabilistic Modeling. Due to the prevalence of uncertain and noisy data in modern applications, probabilistic models have been extensively studied for probabilistic databases [50], information extraction [59], and data integration [1]. Sen and Deshpande [50] introduce graphical models to represent and query correlated tuples (e.g., independence and implication relationships) efficiently, framing query evaluation as an inference problem over probabilistic graphical models. However, their approach relies on exact inference, limiting their scalability to smaller graphs with tens of thousands of random variables as opposed to millions of variables in our case. Wang et al. [59] integrate Conditional Random Fields (CRFs) into relational query processing, demonstrating how probabilistic inference improves both the accuracy and efficiency of querying uncertain information extracted from unstructured sources. Their approach focuses on tasks such as named entity recognition and sequence labeling, where the linear structure of input text sequences makes linear-chain CRFs a natural fit. In contrast, our integration problem requires a more general graph structure to model the transitivity of ternary concept groups. Consequently, their CRF-based techniques and optimizations, designed for linear sequences, do not directly apply to our setting. Agrawal et al. [1] establish the theoretical foundations for uncertain data integration within the Local-As-View framework, introducing novel containment notions—equality-containment and superset-containment—to model how uncertain data from diverse sources can be consistently merged. While their theoretical framework is sound, the paper leaves the development of efficient query processing techniques and real-world applications as open areas for future work.

Metadata and Dataset Discovery. Metadata has been extensively used in facilitating downstream tasks such as data discovery and data sharing. Google dataset search indexes the metadata of datasets (published by users in `schema.org` format) [8]. Auctus, a domain-specific and open-source dataset search engine, supports spatial and temporal data augmentation through indexing the data summaries describing dataset contents (e.g. grid size for spatial data) [10]. Raw metadata (e.g., column names and tags) [5, 41], concept annotations [37], and concept hierarchies [27] have been used as data summaries to be indexed for table union and join search. Besides data cataloging, the hierarchies inferred by OpenForge can further enhance the results quality of downstream data discovery tasks, particularly for domain-specific data repositories. For instance, datasets tagged with metadata of varying granularity could better align with user queries, improving retrieval accuracy [41, 45]. Exploring these applications represents a promising direction for future work.

9 CONCLUSION

We introduce the metadata integration problem and propose OpenForge, a data-driven solution that unifies metadata concepts for a given relationship. By combining advanced prior models with probabilistic modeling and inference using Markov Random Fields, OpenForge effectively resolves relationships between metadata concepts while enforcing essential relationship properties such as transitivity. Our approach also demonstrates both efficiency and scalability across various datasets. A promising direction for future work is leveraging integrated metadata vocabularies to facilitate the discovery of siloed datasets.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants 1946932 and 2107050, a Rackham Predoctoral Fellowship, and Advanced Research Computing at the University of Michigan, Ann Arbor.

REFERENCES

[1] Parag Agrawal, Anish Das Sarma, Jeffrey D. Ullman, and Jennifer Widom. 2010. Foundations of Uncertain-Data Integration. *Proc. VLDB Endow.* 3, 1 (2010), 1080–1090.

[2] Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. 2023. Gemini: A Family of Highly Capable Multimodal Models. *CoRR* abs/2312.11805 (2023).

[3] Ankur Ankan and Abinash Panda. 2015. pgmpy: Probabilistic graphical models using python. In *Proceedings of the 14th Python in Science Conference (SCIPY 2015)*. Citeseer.

[4] Mohit Bansal, David Burkett, Gerard de Melo, and Dan Klein. 2014. Structured Learning for Taxonomy Induction with Belief Propagation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*. The Association for Computer Linguistics, 1041–1051.

[5] Alex Bogatu, Alvaro A. A. Fernandes, Norman W. Paton, and Nikolaos Konstantinou. 2020. Dataset Discovery in Data Lakes. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*. IEEE, 709–720.

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2017. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguistics* 5 (2017), 135–146.

[7] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. 2018. *JAX: composable transformations of Python+NumPy programs*. http://github.com/google/jax

[8] Dan Brickley, Matthew Burgess, and Natasha F. Noy. 2019. Google Dataset Search: Building a search engine for datasets in an open Web ecosystem. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. ACM, 1365–1375.

[9] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

[10] Sonia Castelo, Rémi Rampin, Aécio S. R. Santos, Aline Bessa, Fernando Chirigati, and Juliana Freire. 2021. Auctus: A Dataset Search Engine for Data Discovery and Augmentation. *Proc. VLDB Endow.* 14, 12 (2021), 2791–2794.

[11] CKAN. 2005. *The world's leading open source data management system.* https://ckan.org Accessed: 2025-2-2.

[12] Web Data Commons. 2012. Extracting Structured Data from the Common Crawl. http://webdatacommons.org Accessed: 2024-11-29.

[13] Tianji Cong, James Gale, Jason Frantz, H. V. Jagadish, and Çagatay Demiralp. 2023. 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.

[14] Hong Hai Do and Erhard Rahm. 2002. COMA - A System for Flexible Combination of Schema Matching Approaches. In *Proceedings of 28th International Conference on Very Large Data Bases, VLDB 2002, Hong Kong, August 20-23, 2002*. Morgan Kaufmann, 610–621.

[15] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The Llama 3 Herd of Models. *CoRR* abs/2407.21783 (2024).

[16] Gaspard Ducamp, Christophe Gonzales, and Pierre-Henri Wuillemin. 2020. aGrUM/pyAgrum : a toolbox to build models and algorithms for Probabilistic Graphical Models in Python. In *International Conference on Probabilistic Graphical Models, PGM 2020, 23-25 September 2020, Aalborg, Hotel Comwell Rebild Bakker, Skørping, Denmark (Proceedings of Machine Learning Research)*, Vol. 138. PMLR, 609–612.

[17] Stefano Ermon. 2023. CS 228 - Probabilistic Graphical Models. https://ermongroup.github.io/cs228-notes/. Accessed: 2024-3-16.

[18] Stuart Geman and Donald Geman. 1984. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Anal. Mach. Intell.* 6, 6 (1984), 721–741.

[19] Amir Globerson and Tommi S. Jaakkola. 2007. Fixing Max-Product: Convergent Message Passing Algorithms for MAP LP-Relaxations. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*. Curran Associates, Inc., 553–560.

[20] Schema.org Community Group and Steering Group. 2011. Schema.org. https://schema.org. Accessed: 2023-10-23.

[21] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017 (Proceedings of Machine Learning Research)*, Vol. 70. PMLR, 1321–1330.

[22] Alon Y. Halevy, Flip Korn, Natalya Fridman Noy, Christopher Olston, Neoklis Polyzotis, Sudip Roy, and Steven Euijong Whang. 2016. Goods: Organizing Google's Datasets. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*. ACM, 795–806.

[23] Marti A. Hearst. 1992. Automatic Acquisition of Hyponyms from Large Text Corpora. In *14th International Conference on Computational Linguistics, COLING 1992, Nantes, France, August 23-28, 1992*. 539–545.

[24] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

[25] ICPSR. 1962. *ICPSR Subject Thesaurus.* https://www.icpsr.umich.edu/web/ICPSR/thesaurus/10001 Accessed: 2023-10-23.

[26] ICPSR. 1962. *Inter-university Consortium for Political and Social Research.* https://www.icpsr.umich.edu/web/pages/about/ Accessed: 2023-10-23.

[27] Aamod Khatiwada, Grace Fan, Roee Shraga, Zixuan Chen, Wolfgang Gatterbauer, Renée J. Miller, and Mirek Riedewald. 2023. SANTOS: Relationship-based Semantic Table Union Search. *Proc. ACM Manag. Data* 1, 1 (2023), 9:1–9:25.

[28] Daphne Koller and Nir Friedman. 2009. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press.

[29] Keti Korini, Ralph Peeters, and Christian Bize. 2023. *SOTAB V2 - Table Annotation Benchmark.* http://webdatacommons.org/structureddata/sotab/v2/ Accessed: 2025-2-27.

[30] Keti Korini, Ralph Peeters, and Christian Bizer. 2022. SOTAB: The WDC Schema.org Table Annotation Benchmark. In *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching, SemTab 2021, co-located with the 21st International Semantic Web Conference, ISWC 2022, Virtual conference, October 23-27, 2022 (CEUR Workshop Proceedings)*, Vol. 3320. CEUR-WS.org, 14–19.

[31] Christos Koutras, George Siachamis, Andra Ionescu, Kyriakos Psarakis, Jerry Brons, Marios Fragkoulis, Christoph Lofi, Angela Bonifati, and Asterios Katsifodimos. 2021. Valentine: Evaluating Matching Techniques for Dataset Discovery. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021*. IEEE, 468–479.

[32] Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. NV-Embed: Improved Techniques for Training LLMs as Generalist Embedding Models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

[33] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[34] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep Entity Matching with Pre-Trained Language Models. *Proc. VLDB Endow.* 14, 1 (2020), 50–60.

[35] Marius Lindauer, Katharina Eggensperger, Matthias Feurer, André Biedenkapp, Difan Deng, Carolin Benjamins, Tim Ruhkopf, René Sass, and Frank Hutter.

2022. SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization. *J. Mach. Learn. Res.* 23 (2022), 54:1–54:9.

[36] Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Cristian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, and et al. 2024. Gemma: Open Models Based on Gemini Research and Technology. *CoRR* abs/2403.08295 (2024).

[37] Renée J. Miller, Fatemeh Nargesian, Erkang Zhu, Christina Christodoulakis, Ken Q. Pu, and Periklis Andritsos. 2018. Making Open Data Transparent: Data Discovery on Open Data. *IEEE Data Eng. Bull.* 41, 2 (2018), 59–70.

[38] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep Learning for Entity Matching: A Design Space Exploration. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*. ACM, 19–34.

[39] Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help?. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. 4696–4705.

[40] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy Belief Propagation for Approximate Inference: An Empirical Study. In *UAI '99: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, Stockholm, Sweden, July 30 - August 1, 1999*. Morgan Kaufmann, 467–475.

[41] Fatemeh Nargesian, Ken Q. Pu, Bahar Ghadiri Bashardoost, Erkang Zhu, and Renée J. Miller. 2023. Data Lake Organization. *IEEE Trans. Knowl. Data Eng.* 35, 1 (2023), 237–250.

[42] Fatemeh Nargesian, Erkang Zhu, Renée J. Miller, Ken Q. Pu, and Patricia C. Arocena. 2019. Data Lake Management: Challenges and Opportunities. *Proc. VLDB Endow.* 12, 12 (2019), 1986–1989.

[43] Fatemeh Nargesian, Erkang Zhu, Ken Q. Pu, and Renée J. Miller. 2018. Table Union Search on Open Data. *Proc. VLDB Endow.* 11, 7 (2018), 813–825.

[44] OpenAI. 2023. GPT-4 Technical Report. *CoRR* abs/2303.08774 (2023).

[45] Paul Ouellette, Aidan Sciortino, Fatemeh Nargesian, Bahar Ghadiri Bashardoost, Erkang Zhu, Ken Q. Pu, and Renée J. Miller. 2021. RONIN: Data Lake Exploration. *Proc. VLDB Endow.* 14, 12 (2021), 2863–2866.

[46] Judea Pearl. 1989. *Probabilistic reasoning in intelligent systems - networks of plausible inference*. Morgan Kaufmann.

[47] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12 (2011), 2825–2830.

[48] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67.

[49] Stephen Roller, Douwe Kiela, and Maximilian Nickel. 2018. Hearst Patterns Revisited: Automatic Hypernym Detection from Large Text Corpora. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*. Association for Computational Linguistics, 358–363.

[50] Prithviraj Sen and Amol Deshpande. 2007. Representing and Querying Correlated Tuples in Probabilistic Databases. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*. IEEE Computer Society, 596–605.

[51] Glenn Shafer and Prakash P. Shenoy. 1990. Probability propagation. *Ann. Math. Artif. Intell.* 2 (1990), 327–351.

[52] Jiaming Shen and Jiawei Han. 2022. *Automated Taxonomy Discovery and Exploration*. Springer.

[53] Roee Shraga, Avigdor Gal, and Haggai Roitman. 2020. ADnEV: Cross-Domain Schema Matching using Deep Similarity Matrix Adjustment and Evaluation. *Proc. VLDB Endow.* 13, 9 (2020), 1401–1415.

[54] Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2006. Semantic Taxonomy Induction from Heterogenous Evidence. In *ACL*.

[55] Socrata. 2007. *The Socrata Open Data API*. https://dev.socrata.com Accessed: 2025-2-2.

[56] Technology Transformation Services The U.S. General Services Administration. 2009. Data.gov. https://opendata.cityofnewyork.us/overview/. Accessed: 2023-10-23.

[57] Andrea K. Thomer, Dharma Akmon, Jeremy York, Allison R. B. Tyler, Faye Polasek, Sara Lafia, Libby Hemphill, and Elizabeth Yakel. 2022. The Craft and Coordination of Data Curation: Complicating Workflow Views of Data Science. *Proc. ACM Hum. Comput. Interact.* 6, CSCW2 (2022), 1–29.

[58] Jianhong Tu, Ju Fan, Nan Tang, Peng Wang, Guoliang Li, Xiaoyong Du, Xiaofeng Jia, and Song Gao. 2023. Unicorn: A Unified Multi-tasking Model for Supporting Matching Tasks in Data Integration. *Proc. ACM Manag. Data* 1, 1 (2023), 84:1–84:26.

[59] Daisy Zhe Wang, Michael J. Franklin, Minos N. Garofalakis, and Joseph M. Hellerstein. 2010. Querying Probabilistic Information Extraction. *Proc. VLDB Endow.* 3, 1 (2010), 1057–1067.

[60] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data* 3, 1 (2016), 1–9.

[61] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 Technical Report. *CoRR* abs/2407.10671 (2024).

[62] Qingkai Zeng, Yuyang Bai, Zhaoxuan Tan, Shangbin Feng, Zhenwen Liang, Zhihan Zhang, and Meng Jiang. 2024. Chain-of-Layer: Iteratively Prompting Large Language Models for Taxonomy Induction from Limited Examples. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, CIKM 2024, Boise, ID, USA, October 21-25, 2024*. ACM, 3093–3102.

[63] Meihui Zhang, Marios Hadjieleftheriou, Beng Chin Ooi, Cecilia M. Procopiuc, and Divesh Srivastava. 2011. Automatic discovery of attributes in relational databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011*. ACM, 109–120.

[64] Guangyao Zhou, Nishanth Kumar, Miguel Lázaro-Gredilla, Shrinu Kushagra, and Dileep George. 2022. PGMax: Factor Graphs for Discrete Probabilistic Graphical Models and Loopy Belief Propagation in JAX. *CoRR* abs/2202.04110 (2022).