Embedding-based Deduplication of Knowledge Graphs using Graph Neural Networks

Emma Pinckers[1], Yuliya Shapovalova[2], Shervin Mehryar*[1] and Michel Dumontier[1]

[1]Institute of Data Science, Maastricht University, Paul-Henri Spaaklaan 1, 6229 GT, Maastricht, Netherlands
[2]Radboud University, Toernooiveld 212, 6525 EC, Nijmegen, Netherlands

Abstract

Knowledge graphs (KGs) built from multiple sources often contain duplicated entities caused by inconsistent naming, differing schemas, and incomplete updates, which reduce their reliability in applications such as research and decision making for life sciences and health care. Traditional deduplication approaches perform reasonably well on simple graphs but struggle to handle the scale and relational diversity of modern KGs. This paper explores how a Relation Graph Convolutional Network (R-GCN) can overcome these limitations by learning from both the structure and semantics of heterogeneous relations. We train an R-GCN model and demonstrate its performance at various levels of scale and diversity in data. Through experimentation, we show that the proposed approach outperforms baseline models on both general purpose and clinical deduplication tasks. Over clinical datasets, the approach is further shown to be reliable and consistent using uncertainty quantification metrics.

Keywords

Relational Graph Neural Networks, Knowledge Graph Embeddings, Deduplication, Scientific & Health Care Data, Uncertainty Quantification, Data Interoperability

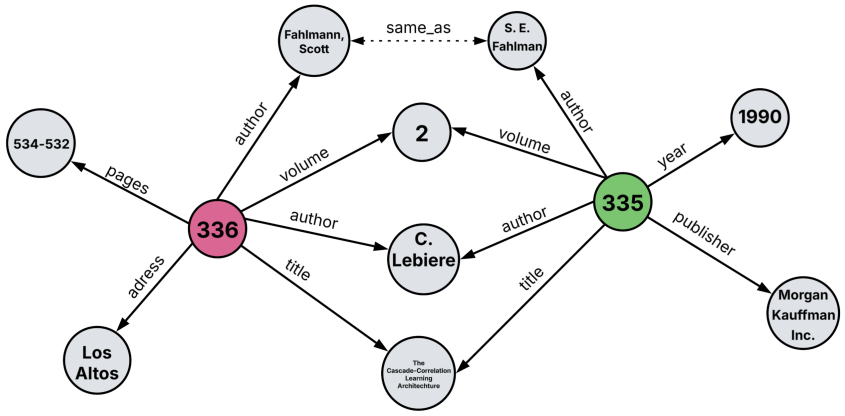
1. Introduction

In recent years, ontology-based integration of data sources whereby properties and inter-connections are represented according to a common terminology, has lead to widespread applications for knowledge graphs [1]. A knowledge graph can be defined as the practical representation of a source knowledge, representing context in a structured way using ontological concepts [2]. Knowledge graphs are used extensively in various research fields such as search engines [3], knowledge management for businesses [4], and health care [5, 6]. To construct and integrate a knowledge graph in practical settings, data needs to be collected from different and disparate sources [7]. These sources often conform to different schemas, contain inconsistencies and are not properly maintained [8, 9].

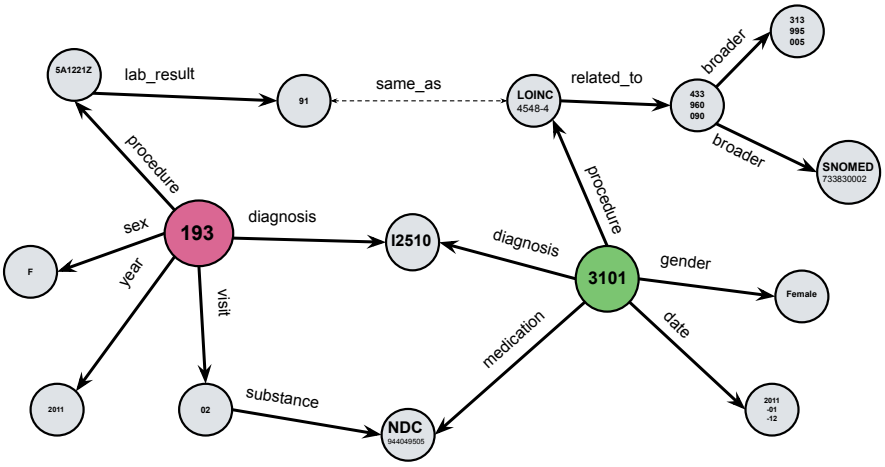
Deduplication refers to the process of resolving identifiers with the goal of discovering nodes within an integrated knowledge graph that refer to the same real-world entity. Figure 1a illustrates this problem. Here two records from the Cora knowledge graph are shown, which is a knowledge graph containing information about various scientific publications. As can be seen in Figure 1a, the records with ID 336 and ID 335 refer to the same paper as they share the title and author. However, they use different naming conventions for the second author (S.E Fahlman and Fahlmann, Scott), as well as provide new information such as the publication year and number of pages. Because of these differences, knowledge graphs often suffer from high duplication rates which requires further resolution. Similarly, across personal health knowledge graphs as shown in Figure 1b, it can be that the same entity (i.e. the same lab result for Hemoglobin A1c in Blood with internal ID 91 as LOINC 4548-4) in one system requires further post-processing in order to determine its representation in another system for a given patient. Although this process can be performed manually, it is tedious and inefficient, especially for large and diverse knowledge graphs. Therefore, an accurate and efficient way

SWAT4HCLS 2026: The 16th International Conference on Semantic Web Applications and Tools for Health Care and Life Sciences, March 23–26, 2026, Amsterdam, Netherlands
*Corresponding author.
shervin.mehryar@maastrichtuniversity.nl (S. Mehryar*)
© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

of performing deduplication is highly desirable for knowledge graph-based applications.



(a) A duplicate pair example from the Cora knowledge graph.



(b) A duplicate pair example from the EHR knowledge graph.

Figure 1: Duplicate pair examples from general purpose (top) and clinical knowledge (bottom) graphs. The 'same as' relation is used to signify a resolution from the proposed approach.

Traditionally, approaches that attempt to solve deduplication tasks rely on blocking and similarity matching [10], [11]. While most of these techniques perform well on simpler knowledge graphs with limited attributes, they often face difficulties when applied to large-scale knowledge graphs containing multiple heterogeneous edge types and complex structures [12], [11]. More recently, neural representation learning is proposed in this area for extracting rich information based on the underlying and graphical structure of the data [5]. The relational graph convolutional network (R-GCN) approach in particular is able to overcome these challenges due to its ability to learn and differentiate between the various edge types in knowledge graphs [13]. This paper aims to investigate the extent to which an R-GCN based approach can improve the accuracy and robustness of knowledge graph duplication detection, especially as the complexity of the graph increases.

Our proposed R-GCN approach focuses on adapting a standard GCN in order to encode the heterogeneous types present within a knowledge graph, as well as, a DistMult decoder [14] trained to identify the occurrence of duplicates. Our methodology is further outlined in Section 2. The model is trained and tested against existing methods on different datasets ranging from scholarly data to personal health knowledge graphs, described in Section 3. The experimentation results are presented

in Section 4, including results and analysis. We conclude our findings in Section 5 by highlighting the benefits of GNN-based embedding of knowledge graphs for the task of de-duplication and provide potential limitations.

2. Methodology

This section details the proposed R-GCN approach for data deduplication of knowledge graphs, which relies on an adapted graph neural network architecture capable of working with multi-layered edges. The R-GCN is designed to solve an intra-graph link prediction task aimed at identifying duplicate nodes in a knowledge graph. In addition to the pre-existing intra-graph edges, the model introduces a new relation type, termed '*sameAs*', to connect matching nodes. This relation is only added between nodes referring to exact or near-exact duplicate records. Embeddings are learned for various node and edge types using the R-GCN encoder, after which a binary classifier in the decoder predicts whether a '*sameAs*' edge exists between a given pair of nodes.

2.1. Problem description

Formally this task can be described as follows: Let $G = (V, E)$ be a knowledge graph, where V is the set of nodes, each representing an entity or object. Each node $v \in V$ is associated with a set of attributes $\{A_1, \ldots, A_m\}$, where $v[A_i]$ denotes the value of attribute A_i for node v. E is the set of edges representing relationships between nodes. The task is, given all distinct node pairs (v, v') from V, where $v \neq v'$, to predict which pairs of nodes correspond to duplicate representations of the same real-world entity.

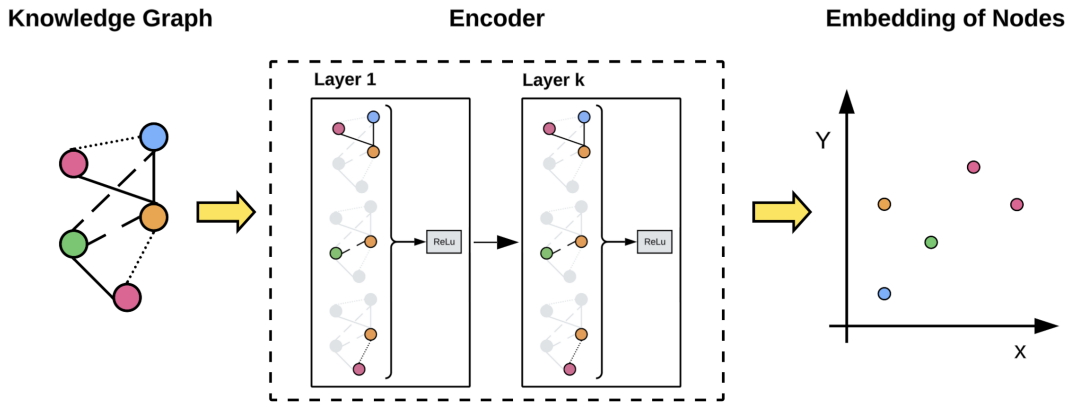
2.2. R-GCN Encoder Design

The R-GCN encoder, as introduced by Schlicht [13], works similarly to a graph convolutional network (GCN) operating on local graph neighborhoods; however, it is explicitly designed to deal with highly multi-relational data typical of realistic knowledge bases. By learning structural representations in relation to the different edge types, it is better equipped to handle different edge types present within a graph in comparison to a standard GCN encoder, making it ideal for working with large-scale multi-relational ontologies. The encoder differs from a traditional GCN encoder in how neighborhood sampling is performed. For multi-relational graphs, each vertex $\vec{v}$ has r neighborhoods, denoted by $\mathcal{N}_r(v)$, for each edge type r. The R-GCN encoder learns the vector representation of $\vec{v}$ by aggregating the incoming relation-dependent messages calculated over a summation of all nodes $u \in \mathcal{N}_r(v)$. This process is repeated for each of the neighborhoods $\vec{v}$ for all different edges types. The input messages, represented as vectors, are subsequently combined and processed using an element-wise activation function. This forward propagation is then repeated for k iterations, resulting in a k-layer R-GCN encoder. The working of the encoder is illustrated in Figure 2a.

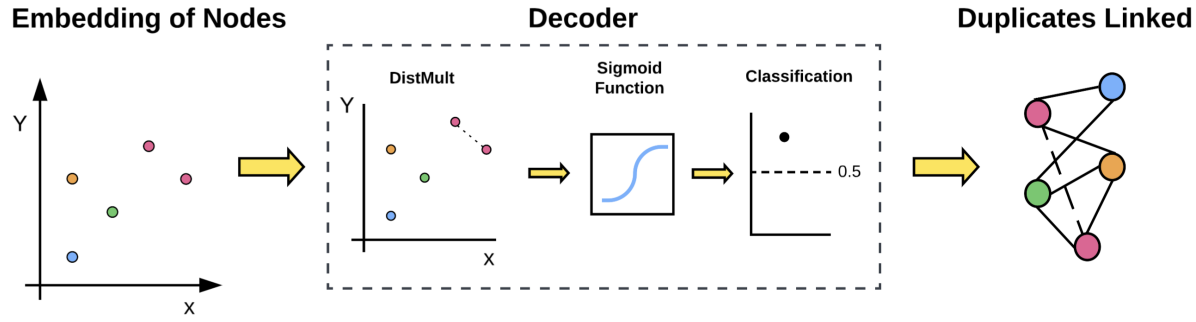
2.3. Decoder Design

The decoder employs a DistMult function which computes a score representing the likelihood of a *sameAs* relationship r existing between two nodes, head h and tail t. This score is calculated as the ternary dot product of the head, relation and tail vectors, expressed as $score(h, r, t) = \sum_{i=1}^{d} h_i \cdot r_i \cdot t_i$, where d is the number of dimensions. Since the aim is to predict whether relation r exists between h and t, directionality in this case does not matter, given that if two nodes are deemed to have a *sameAs* relation r, r is bidirectional by nature. Therefore, the score remains symmetric such that $score(h, r, t) = score(t, r, h)$. A sigmoid activation function, $\sigma(x) = 1/(1 + e^{-x})$, is consequently applied to the calculated score, after which it is compared against a predefined threshold of 0.5 to classify the relationship. If the sigmoid of the score meets or exceeds this threshold, the head and tail nodes are

matched. Figure 2b visualizes the workings of the decoder.



(a) Illustration of the encoder design which looks at all the present edge types to make embeddings.



(b) Illustration of the decoder which uses a DistMult function to predict the existence of a 'sameAs' relation between nodes.

Figure 2: Overview of the proposed R-GCN frame work for KG Deduplication.

2.4. Negative Sampling

Negative edges are introduced to stabilize the training of the R-GCN. These edges are added at the same rate as positive edges, meaning that for each positive edge, there exists a negative edge to ensure balance in the diversity of the nodes and prevent over-representation of certain edges. Given that $h, t \in G$, negative edges are generated by corrupting either the head or the tail node using a node $v \in G$, where $v \neq h, t$.

3. Experimentation

The performance of the model is evaluated using a diverse set of datasets varying in complexity, specifically selected with the aim to examine how the model performs in real-world application scenarios. The datasets vary in terms of size, properties, and noise, and can be divided into 2 categories: general and clinical. All datasets are representative of potential real-word use cases. While it is possible to evaluate performance over synthetic datasets, real-world data is preferable over synthetic data, as it is difficult to simulate all types of errors that might occur during data entry or data processing [15]. A summary of the main characteristics of the datasets can be found in Table 1. In each case, a number of ground truth duplicate exist for training purposes. Performance of the model is evaluated based on industry standard evaluation metrics and compared to three existing solutions for deduplication.

Additionally, the computation time and data efficiency are taken into account to assess the efficiency of the model.

Dataset	Complexity	Triples	Attributes	Total Entries	Original Entries	Duplicate Pairs
General Datasets						
KGDL	Low	904	2	1277	1090	187
Countries	Medium	1569	4	523	300	229
Cora	High	105384	17	1879	182	64578
Clinical Datasets						
elCU-phkg	High	13930	4	3317	717	1153
EHR	Medium	60328	16	138	100	40
phkg-hadm	High	19482	4	9527	1537	2370

Table 1
Datasets and their characteristics. Total Entries refers to the number of records in the dataset. Original Entries are the records that remain if all duplicates are removed. Duplicate Pairs are the total links that need to be identified.

3.1. General Datasets

3.1.1. The KGDL dataset

The KGDL (Knowledge Graph for Deduplication in Literature) dataset is a structured RDF-based collection of bibliographic metadata sourced from academic repositories such as arXiv, Google Scholar, and Zotero. It captures scholarly articles, preprints, and conference papers, detailing titles, authors, abstracts, identifiers (e.g., DOIs, URLs), subjects, and attachments (e.g., PDFs, snapshots). The dataset emphasizes relationships, using namespaces like Dublin Core, FOAF, and Zotero to link publications to authors, publishers, and attachments, with deduplication logic identifying and resolving redundant entries through owl:sameAs relationships, ensuring a clean knowledge graph for literature management. This dataset has the lowest complexity and smallest size among the datasets used to evaluate the model.

3.1.2. The Countries Dataset

The second dataset called '*Countries*' was originally developed by Garshol and Borge [10], with the goal of evaluating their proposed deduplication tool, '*DuplicateKiller*' (Duke). The dataset consists of a knowledge graph constructed from records from the DBPedia [16] and the Mondial database [17]. The knowledge graph contains structured geographical information, such as total area and capital city, about various countries and territories. In total the dataset contains 523 records, of which 300 are unique entries, leaving 223 duplicates. Each original instance has either zero, one or at most two duplicate entries (6 cases), resulting in approximately 229 duplicate links that must be correctly identified. The performance of the R-GCN on this dataset is compared against that of Duke on the same dataset. Duke was originally developed in 2014 by Garshol and Borge while working on a project for Hafslund, a Norwegian energy company. While Duke is a relatively old model for deduplication, it is still considered one of the most effective tools for deduplication[18]. It is a non-machine learning approach to deduplication, which uses a rule-based probabilistic model, specifically Naive Bayesian inference, to determine if two records refer to the same real-life entity. Using Bayes' Theorem and the Naive Bayes assumption, the model computes $P(M|e)$ for each candidate pair. Here, M denotes the event that the two records match, i.e. represent the same entity. $e = \{e_1, \ldots, e_n\}$ refers to the observed similarities between n properties, also referred to as fields, of the records. Each $e_i \in [0, 1]$ is the result of a field-level comparison function. In order to avoid full pairwise comparisons, Duke uses Lucene-based /cite indexing to efficiently select candidate pairs. The decision rule for a match is: $Match\ if\ P(M \mid e) \geq \theta$, where $\theta \in [0, 1]$ is set to a predefined threshold. For the Countries dataset this

threshold was set to 0.63 via grid search.

3.1.3. The Cora Dataset

The final general dataset termed 'Cora' was developed by Draisbach and Naumann [15] as part of an effort to build a standard benchmark for testing deduplication models. In the associated paper, three datasets built from real-world data are introduced, varying in size and complexity. The evaluation of the R-GCN focuses on its performance on the most complex dataset, Cora. Originally Cora is a citation network of machine-learning papers, developed by Andrew McCallum [19], with the intent to be used as a dataset for classification tasks. It contains attributes of academic papers such as authors, title and date. Draisbach and Naumann adapted the dataset to make it easily usable for deduplication tasks by performing normalisation steps and by providing a ground truth, listing all the duplicates present in the data. In total, the Cora dataset contains 1879 records, of which 182 are considered original entries. Among these entries, 64 records are completely original, occurring only once, while the remaining records exhibit a widely varying number of occurrences, i.e. duplicates. Notably, the most recurring record totals 238 appearances, a significant increase compared to Countries. In total, the dataset itself requires identification of approximately 64,000 duplicate links. This number results from the transitive nature of the data, since if record ID_1 has duplicates record ID_2 and ID_3, the implication is that ID_2 and ID_3 are duplicates as well. This combinatorial property introduces a significant amount of noise and complexity to the data. Additionally, the dataset contains up to 17 different attributes per record, resulting in a wide variety of different edge-types being used, adding to the overall complexity of the dataset. An example of a duplicate pair is illustrated in Figure 1a . The performance of the R-GCN is compared to a recent approach by Sohail and Qounain [11] termed Locality Sensitive Blocking or LSB. LSB uses locality sensitive hashing to group together similar records into blocks, pairs grouped into the same block are called candidate pairs. This is done to avoid having to full pair-wise comparisons across all records. After the blocking steps, the final candidate pairs are compared using similarity metrics like Jaccard similarity to determine if they are true duplicates.

3.2. Clinical Datasets

The following datasets are the results of graphs generated as an RDF knowledge graph derived for clinical outcome prediction from multiple data sources (originally in tabular format) by the resolution-alignment-completion (RAC) system proposed in [20]. The graphs are generated using intensive care unit records across multiple hospitals based on different schemas. Because patients, admissions, procedures, lab events, and medication records can appear redundantly in different segments of the graph, identity links are used to merge and establish equivalence. The resulting graph provides a unified, ontology-driven representation of patient records after deduplication. Complexity arises from the highly interconnected structure as each patient may be linked to numerous events, as explained below in each case.

3.2.1. The eICU-phkg Dataset

This subset is derived from a single clinical source containing duplicate representations of diagnosis, lab, and medication codes. Each local code in the dataset is aligned with a reference ontology and can correspond to several SNOMED CT identifiers. The deduplication task consists of identifying when multiple local codes refer to the same clinical concept and consolidating those equivalence clusters into canonical forms. Although the structure is small, complexity arises from the multiplicity and overlap of these mappings: a single local identifier can map to several SNOMED concepts. This produces equivalence sets that partially intersect, propagate through subclass relations, or converge on shared SNOMED ancestors.

3.2.2. The EHR Dataset

In this subset of the RDF data, 60328 triples are contained describing 138 patient-related entries drawn from two different systems with disparate schemas. Although a large number of triples are present, a small set of attributes (five) are present connecting a dense collection of ICD stays, lab results and medication records. The long chain of linked entities creates structural complexity whereby a single duplicate patient may be embedded based on a proportionally large number of related entities, making it non-trivial to identify and collapse redundant representations. Deduplication is established by linking equivalent patient nodes.

3.2.3. The phkg-hadm Dataset

This subset contains identifiers for diagnosis codes, lab test codes, and medication codes that appear under multiple namespaces but represent the same underlying clinical concepts. It also contains a reference ontology, namely SNOMED CT, to align multiple entities, which ties together many-to-one mappings between local EHR toolkit codes and SNOMED CT identifiers. For example, a single ICD code or lab item code may be declared equivalent to several SNOMED concepts, producing clusters of co-referent entities that must be collapsed into canonical forms. The complexity comes from the structure of the biomedical ontology itself as well as their broader taxonomic relationships.

3.3. Configurations

As can be observed in Table 1 the datasets vary greatly in complexity and size. This results in the need to configure the embedding dimensions for the R-GCN. Larger embedding dimensions are usually ideal for capturing complex patterns within datasets but carry a high risk of over-fitting, resulting in poor generalizations. As a result, each task requires different embedding dimensions to achieve optimal performance. To determine the ideal embedding dimensions for each task, the model is trained on the training data of the task in question and hyper-parameters, only altering the embedding dimension, after which performance is assessed using a validation set. The same process was also performed to choose the optimal Number of Layers for each dataset, as graphs with important long-range dependencies often need more layers to successfully propagate meaningful features across the graph. Similarly, the Learning Rate was adjusted for each dataset to ensure an appropriate balance between training stability and convergence speed. The chosen optimal embedding dimensions, Number of Layers, and Learning Rate are shown in Table 2.

For both experiments, the R-GCN encoder implements two RGC operators, each followed by a ReLu activation and a Dropout layer with a rate of ($p = 0.2$) using the PyGeometric library [21]. The hyper-parameters are optimized using a validation split and set to `batch_size=1024`, `dropout=0.2`, `regularization=1e-2`. For all datasets the R-GCN was trained for 100 epochs.

Dataset	Embedding Dimension	Number of Layers	Learning Rate
KGDL	200	3	5e-3
Countries	400	5	5e-3
Cora	300	3	2e-3
eICU-phkg	200	4	5e-4
EHR	200	3	5e-3
phkg-hadm	800	3	2e-4

Table 2
Datasets and their Embedding Dimension, Number of Layers, and Learning Rate.

3.4. Performance Evaluation

The performance of the R-GCN on the datasets is evaluated using standard metrics, namely, Recall Precision and F1-Score to determine the correctness of the predictions made, defined as follows. Here, D_{out} refers to the duplicates links predicted by the model, while D_{ref} refers to the ground truth duplicate links.

$$P = \frac{|D_{out} \cap D_{ref}|}{|D_{out}|}, \qquad R = \frac{|D_{out} \cap D_{ref}|}{|D_{ref}|}, \qquad F = \frac{2PR}{P+R} \tag{1}$$

Mean Rank and Hits@k are further utilised to show how accurately each model can rank its predictions. The performance is evaluated at both $k = 1$ and $k = 10$.

$$Hits@K = \frac{|\{d \in D_{ref} | Rank(d) \leq K\}|}{|D_{ref}|}, \qquad MR = \frac{\sum_{d \in D_{ref}} Rank(d)}{|D_{ref}|} \tag{2}$$

For the comparison between the performance of the R-GCN and the previously described models, only precision, recall, and F1-score are taken into account. This is done since Mean Rank and Hits@k are were not provided by the baseline models and are thus only evaluated for the R-GCN. In addition to standard evaluation metrics, computation time and data efficiency are also measured. This is done to assess the feasibility and efficiency of each model in practical deployment scenarios. While accuracy-based metrics are critical for evaluating model effectiveness, runtime performance plays an equally important role, particularly in real-world applications where scalability and response time are key operational concerns. All experiments were performed using Tesla T4 GPU.

4. Results and Discussion

As can be observed in Table 3, the R-GCN showcases great results in terms of its average performance, performing well across all datasets. The R-GCN achieves an average score of 0.96 and 0.96 for both recall and precision, resulting in a mean F1-score of 0.96 as well. This showcases the R-GCN's accuracy and precision in detecting duplicates, correctly identifying all known duplicate pairs without error. Additionally, the model has a similarly strong performance when evaluated on Mean Rank and Hits@1. The model is able to attain a Mean Rank of 0.541 and Hits@1 score of 0.734. While these results are not as perfect in comparison to the F1-score, this discrepancy is minimal, and the model still showcases the ability to accurately rank duplicate matches. However, ultimately, the complexity and structure of the task also play a role in the performance of the R-GCN, which is illustrated in Table 3, highlighting the differences in performance across the tasks. These differences are further explored in the next subsections.

4.1. Performance on the General Datasets

When looking at the model's performance on the general datasets, it can be seen that the R-GCN achieves near-perfect performance on all three datasets in terms of recall, precision, and F1-score. However, a decline in performance is seen for the ranking-based metrics, Mean Rank and Hits@k, as the complexity of the datasets increases.

4.1.1. Performance on the KGDL Dataset

The best performance of the model is seen on the KGDL dataset, with a Hits@1 of 0.92 and a Mean Rank of 0.08. While these results are still impressive, they are likely achievable by most standard deduplication tools, given the dataset's relatively low complexity and minimal noise.

Dataset	Model	Metrics					
		P	R	F	Hits@1	Hits@10	Mean Rank
General Datasets							
KGDL	R-GCN	1.000	1.000	1.000	0.918	1.000	0.081
Countries	Duke	0.990	0.940	0.965	-	-	-
	R-GCN	1.000	1.000	1.000	0.848	1.000	0.260
Cora	LSB	0.562	0.904	0.693	-	-	-
	R-GCN	0.999	0.999	0.999	0.655	0.997	0.492
Clinical Datasets							
eICU-phkg	S-Graphormer	0.852	0.651	0.738	-	-	-
	R-GCN	0.964	0.963	0.963	0.417	1.000	1.350
EHR	S-Graphormer	0.615	0.533	0.571	-	-	-
	R-GCN	0.938	0.929	0.928	1.000	1.000	0
phkg-hadm	S-Graphormer	0.856	0.682	0.759	-	-	-
	R-GCN	0.875	0.853	0.851	0.563	0.999	1.070
Average overall		**0.963**	**0.957**	**0.957**	**0.734**	**1.000**	**0.541**

Table 3
Model performance on the datasets. Baseline methods LSB and Duke are shown for Countries and Cora, as well as, S-Graphormer for eICU-phkg, EHR, and phkg-hadm. Evaluated using Precision (P), Recall (R), F-1 score (F), Hits@1, Hits@10 and Mean Rank.

4.1.2. Performance on the Countries Dataset

In comparison to its performance on KGDL, the model shows slightly reduced performance on the Countries dataset, obtaining a Hits@1 of 0.85 and a Mean Rank of 0.26. A likely reason for the disparity in the model's performance is the greater structural complexity and number of nodes of the Countries dataset. However, in isolation, the performance of the model is still more than adequate, highlighting that it is still able to work effectively even with increased complexity. This is further confirmed by the fact that the model is able to realize perfect recall, precision, F1 and hits@10.

4.1.3. Performance on the Cora Dataset

The strengths of the model are most evidently shown on the Cora dataset, where it demonstrates a robust performance. Most impressively, the model is able to get a near-perfect, recall, precision, and F1-score, identifying all 64000 duplicate links present in the dataset with little error. When compared against its performance on the previous datasets, a drop in performance is seen in terms of hits@1 and Mean Rank. The model achieves a Hits@1 of 0.65 and a Mean Rank of 0.49. Although these values are suboptimal in absolute terms, they are reasonable given the highly noisy nature of the dataset. These results are especially impressive given the context that in the Cora dataset most records have a large number of duplicates, making it extremely challenging to consistently rank the correct match in the top position. To show that this is the cause of the worsened performance, a new version of the dataset Cora dataset was created. Here, records had at most one other duplicate record, significantly reducing the number of duplicate links that need to be identified. The performance of the model on this dataset is shown in Table 4. As can be seen, the model performs better both in terms of Mean Rank and Hits@1, scoring 0.260 and 0.820 respectively, matching the performance on the Countries dataset. This indicates that the relatively poor ranking performance on Cora is a reflection of the dataset characteristics, rather than and intrinsic issue of the model, as it is nearly impossible to get a good ranking score due to the

nature of the data. Once the number of duplicates per entry is reduced, the R-GCN is able to improve its performance and effectively deal with the multiple-edge types present in Cora.

Dataset	Model	Metrics					
		P	R	F	Hits@1	Hits@10	Mean Rank
Adapted Cora	R-GCN	1.00	1.00	1.00	0.820	1.00	0.259

Table 4
R-GCN performance on the adapted Cora dataset. Evaluated using Precision (P), Recall (R), F-1 score (F), Hits@1, Hits@10, and Mean Rank.

4.2. Performance on the Clinical Datasets

The achieved results on the clinical datasets indicate that the model sustains its high performance when applied to real clinical data, both in terms of duplicate identification accuracy and prediction ranking. On these datasets, the model, on average, obtains an f1-score of 0.91 and Mean Rank of 0.81. These consistently high scores would suggest that the R-GCN architecture is suitable for practical clinical data entity resolution tasks. We later quantify these results in terms of calibration and uncertainty in Section 4.6.

The best overall performance is observed on the EHR dataset, where the model achieves perfect hits@1, hits@10 and a Mean Rank of 0. These results stand out compared to the performance on the phkg-hadm and eICU datasets, where the ranking metrics noticeably lag behind. On these datasets the R-GCN is only able to realize hits@1 scores of around 0.5 and a Mean Rank of 1. While in comparison the results on the EHR dataset are impressive, they reflect characteristics of the dataset rather than an inherent bias of the model: EHR contains very few duplicate candidates per query, which means that for most predictions, there is only a single correct match. Under these conditions, the model can consistently identify and rank the correct duplicate at the top position, producing optimal ranking metrics. Using the same reasoning, the lower ranking performance on eICU and phkg-hadm is expected. Both datasets contain a much larger number of duplicate candidates per entry, averaging 18 and 7 duplicates respectively, substantially increasing the difficulty of ranking the correct match first. As mentioned in section 4.1.3, the same phenomenon was observed on the Cora dataset. Nevertheless, the model is still able to showcase sufficient ranking metrics, consistently placing the correct entity within the top 10 predictions.

Furthermore, the model is able to compensate for its shortcomings in the ranking metrics with its performance on accuracy based metrics. Here, the model performs well across all datasets. It achieves an f1-score of around 0.95 on both the EHR and eICU datasets, indicating a balanced and reliable ability to correctly identify duplicates. The somewhat lower f1-score of 0.85 on the phkg-hadm dataset comes from a slight imbalance between precision, 0.88, and recall, 0.85. A similar discrepancy can be seen on the EHR dataset, however here the difference is only minimal. This disparity suggests that, while the model is generally accurate, it is slightly more conservative as it avoids false positives at the expense of missing some true matches. From a practical perspective, this trade off indicates that the R-GCN prioritizes the correctness of its positive predictions but could benefit from further tuning to improve sensitivity on datasets with high duplicate density such as the eICU dataset.

4.3. Comparison Between Performance on the General and Clinical Datasets

In comparison to the general datasets, the model exhibits a slightly worse performance on the clinical datasets, with differences observed in both ranking quality and identification accuracy. However, the

discrepancy in accuracy is largely negligible as the model still maintains an average f-score of 0.91 in contrast to an average score of 1 on the general datasets, indicating that the model still maintains its accuracy even under more challenging scenarios. The decline in ranking performance however, is more pronounced, marking it as a clear area for potential future improvement, with the notable exception of the EHR dataset on which the model outperforms all other results. The performance on the eICU and phkg-hadm datasets is most similar to the models performance on the Cora dataset. This similarity was to be expected as all three of these datasets have high levels of noise, heterogeneity, and increased structural complexity, largely introduced by the high duplicate count per entry. The effect of this is reflected in the models' achieved results, having a weaker performance on ranking metrics compared to accuracy, as described in previous sections. These observations further suggest that the performance gap is driven primarily by dataset characteristics inherent in real clinical data rather than inherent limitations of the model, highlighting opportunities to further enhance ranking through data preprocessing, model tuning, or architecture refinements.

4.4. Comparison with Existing Approaches

When evaluating the R-GCN's performance in comparison to other deduplication approaches it becomes clear that the model is able to meet the current standards for deduplication. The model is able to outperform the two deduplication approaches described in section 3, named Duke and LSB. Especially in comparison to LSB, the R-GCN is able to highlight its superiority when dealing with noise and complex datasets.

4.4.1. Comparison with Duke

The performance of the model on the Countries dataset was compared to that of Duke on the same dataset. Duke is already able to achieve very high scores in terms of recall and precision, attaining scores of 0.940 and 0.990 respectively. Despite this high performance, the R-GCN is able to narrowly beat the performance of Duke, scoring a 1 in all categories. This shows the R-GCN is able to meet and even exceed current state-of-the-art approaches to deduplication in terms of accurately identifying all duplicates. Since no information regarding ranking-based metrics was provided in the original paper for Duke, the performance of the R-GCN in this aspect can therefore not be evaluated in comparison. For a more complete analysis of the performance of the R-GCN, this type of comparison should be carried out in the future.

4.4.2. Comparison with LSB

In comparison to the LSB method, the R-GCN vastly outperforms the method on the Cora dataset. For precision recall and F1-score, R-GCN is able to improve the performance over LSB by 0.438, 0.096 and 0.307 respectively. This highlights the R-GCN's strength in dealing with large and complex datasets, being able to maintain optimal performance where other models struggle. Especially in terms of precision a big improvement is found, showing the R-GCN is more accurate in its approach compared to LSB. However, it is important to note that precision and F1-score, as well as ranking-based metrics, were not directly reported in the original paper. Instead, the original paper reports on the Reduction Ratio (RR), showing the LSB method is likely optimized for efficiency not accuracy. For comparison with the R-GCN, precision and the F1-score were calculated based on other data presented in the paper, therefore some discrepancies may be present. Similarly to the comparison with Duke, more complete analysis would be needed in the future to accurately assess the performance of the R-GCN compared to that of LSB.

4.4.3. Comparison with S-Graphormer

In order to compare the performance of the proposed method on clinical datasets, we train and benchmark our approach against the Small Graphormer (S-Graphormer) proposed for entity disambiguation

in [5]. The S-Graphormer utilizes a transformer architecture with self-attention in order to process graphical data. It applies a positional encoding scheme using the graph Laplacian that aims to incorporate global and structural information. It varies in architecture from the original Graphormer in that the node distance and centrality feature biases are removed. In this case, we use the default parameters and report only precision, recall, and f1 scores which are more relevant in clinical settings and spare with the ranking metrics. We observe that overall the R-GCN outperforms the S-Graphomer on the deduplication task across all three clinical datasets. Over the EHR dataset, the S-Graphormer is unable to achieve comparable results due to large graph size and limited number of duplicate pairs for training, which cause the global information to overwhelm its predictions. On eICU and phkg-hadm datasets, the R-GCN outperforms the S-Graphormer by 0.22 and 0.09 points in f1 score respectively. This drop in performance as compared to R-GCN is due to the use of positional encoding. The task of deduplication in this case can be performed by local message passing over heterogeneous relation types as done by R-GCN more effectively.

4.5. Computation Time Evaluation

Dataset	Computation Time per Epoch (s)
KGDL	0.39
Countries	1.67
Cora	68.3
eICU-phkg	0.64
EHR	18.2
phkg-hadm	5.73
Average	15.7

Table 5
Computation time per epoch for R-GCN on each dataset.

The performance of the R-GCN in terms of computation time is reported in Table 5. As can be seen, the R-GCN has an average of 15.7 seconds per training epoch. This is a suboptimal result and a significant drawback of the model, especially when training consists of 100 epochs. However, it is worth noting that the average computation time is predominantly affected by the model's runtime on the Cora dataset, where training takes 68.3 seconds per epoch. On the smaller and less complex datasets KGDL, Countries and eICU, the R-GCN significantly cuts down its computation time, requiring 0.39 seconds per epoch for KGDL, 1.67 seconds for Countries, and 0.64 for eICU. This reveals the relation between dataset complexity and computation time. However, even on the smaller datasets, there is still a noticeable area of improvement, particularly when compared to performance of other models like Duke. Duke's accuracy on the Countries dataset might lag slightly behind that of the R-GCN, but in terms of computation time it has the upper hand, only needing around 1.86 seconds in total to execute a task, whereas the R-GCN takes 1.67 seconds per epoch. Although, this discrepancy is minimised given the fact that the R-GCN does not require a large amount of training data to achieve desirable results.

Dataset	0%*		25%*		50%*		75%*		100%*	
	MR	F	MR	F	MR	F	MR	F	MR	F
eICU-phkg	6.02	0.41	5.22	0.76	2.38	0.93	1.74	0.94	1.35	0.95
EHR	11.34	0.33	3.72	0.43	0.00	0.56	0.00	0.85	0.00	0.93
phkg-hadm	7.76	0.44	1.70	0.73	1.23	0.80	1.10	0.82	1.00	0.85
Average	**8.37**	**0.39**	**3.55**	**0.64**	**1.20**	**0.76**	**0.95**	**0.87**	**0.78**	**0.91**

Table 6
R-GCN performance measured by F1-score (F) and Mean Rank (MR) with different percentages of training data available.

This observation can be established from looking at Table 6 which shows how the model's performance changes as more training data becomes available. As expected, the model's performance declines as the amount of training data decreases. At 0% available training data, essentially leaving the model untrained, the performance is severely limited, having an average F1-score of 0.39 and Mean Rank of 8.37. Notably, with even as little as 25% of data available, the model is able to achieve a Mean Rank of 0 on the EHR dataset, matching the performance with 100% of the training data used. Similar behavior can be observed on the eICU dataset, where 50% is enough for the model to be able to match the F1-score of 0.95 it achieved with the full training data. On average, having between 50% and 75% of training data available, will be sufficient for the model to realize impressive results.

These results suggest that the model is highly data efficient, as it is able to show good generalization even when trained on limited datasets. The findings on the ehr and eICU datasets indicate that the model is capable of capturing the underlying relational patterns with relatively little data. This reinforces the idea that the model is not only efficient but also resilient to data sparsity, a desirable property in clinical settings where annotated or labeled data may be limited. Altogether, these results imply that the model's performance does not depend on the sheer volume of training instances. Considering this, it can be concluded that the high computation time of the model can be negated by reducing the volume of training data, as this will significantly speed up the training process while having a minimal impact on the model's performance.

4.6. Uncertainty Quantification

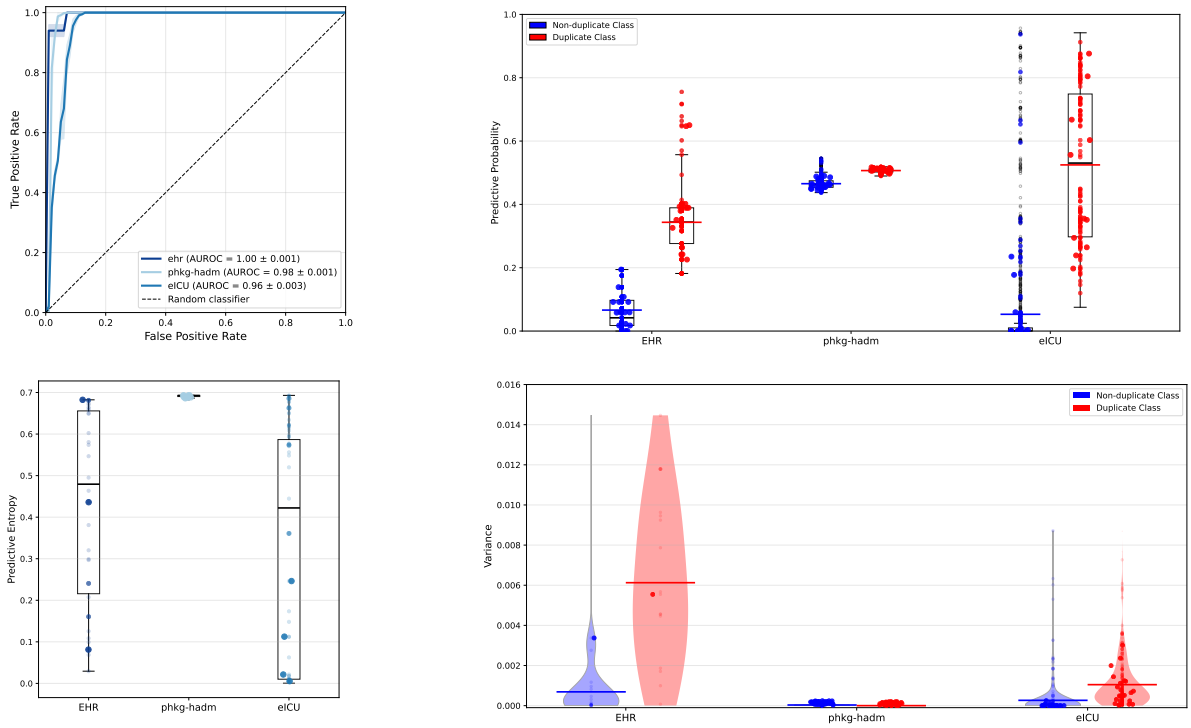


Figure 3: Analysis over the three clinical datasets (EHR, phkg-hadm, and eICU-phkg) in terms of uncertainty via an ensemble of 10 shallow RGCN models ($k = 1$ and drop_out $= 0.2$), namely area-under-the-curves (top-left), predictive probability (top-right), entropy (bottom-left), as well as prediction variances (bottom-right).

Due to the importance of decision making and confidence in clinical applications, in this section the uncertainty in the model's predictions are analyzed and demonstrated in Figure 3. Specifically, we adopt an ensemble setting in which 10 different, shallow RGCN models are trained independently on the training data and utilized in order to make predictions on the test data [22]. We examine and

quantify the performance of the models using a range of metrics. Firstly, the area-under-the-curve in terms of accuracy in this setting is examined and we observe that over the different datasets, the model performs robustly with mean AUROC over 0.96 points. In order to determine the consistency of predictions among the ensemble models, we further plot the prediction values for the duplicate entries (scores above 0.5) versus the non-duplicate entities (scores below 0.5), separately [23]. It can be observed that the models can confidently predict the duplicate class with strong agreement. In the case of EHR and eICU datasets, the predictive probabilities are closer to zero boundary and can cause confusion resulting in a higher false positive rate. The variance for predictions over the two datasets are as such more spread as demonstrated. In the case of phkg-hadm dataset, the predictive variance is low in general, however, the models have the highest entropy (close to 0.69) in this case. This may indicate overconfidence in prediction and explain the concentration of prediction probabilities around 0.5 points.

5. Conclusion

This paper demonstrates that the proposed R-GCN-based approach for deduplication outperforms popular existing methods such as Duke, LSB and S-Graphormer. This is evidenced by multiple evaluation metrics across 6 datasets ranging in complexity. The proposed method uses an adapted GCN encoder design, which, when creating vector embeddings, takes the neighbourhood of a node into account for each of the edge types present in a graph. Based on the embeddings, the decoder utilises DistMult to determine if two nodes should be considered duplicates. This model design allows the model to efficiently deal with complex and heterogenous knowledge graphs, beating other approaches that struggle to perform well on these types of graphs.

The proposed R-GCN performs particularly well in its prediction accuracy, obtaining excellent results for both precision and recall on all 6 datasets, averaging an F-score of 0.957. This is especially impressive for heterogeneous datasets such as Cora and EHR where 17 different edge types are present. On both the clinical and the general datasets, the model matches or exceeds to performance of other state-of-the-art models. The most improvement is found on the EHR dataset where the model is able to increase the F-score of S-Graphormer by 63%. In terms of prediction ranking, the R-GCN performance decreases as the dataset complexity increases, with the exception of the EHR dataset. However, as demonstrated, this is likely the result of the complex datasets having a high duplicate count for each entry, making perfect ranking metrics nearly impossible. Nonetheless, the model still obtains a Hits@10 of 1 on all datasets, suggests that the model still consistently retrieves correct matches within the top range of candidates. The uncertainty analysis of the model showed the model can confidently and consistently predict the duplicates with strong agreement.A possible downside of a R-GCN approach is the computation time. Here, the model falls behind in comparison to recent approaches. This is an area of improvement for the R-GCN model. However, a large computation time is often reasonable for large datasets like EHR and Cora, where precision is likely preferred over speed. Additionally, the model is highly data efficient, in some cases requiring only 50% of the training data to achieve outstanding results. This allows for a significant reduction in the computation time.

In summary, the R-GCN model offers a highly accurate, graph-aware solution to knowledge graph deduplication, with notable advantages over existing methods, particularly in handling relational complexity and achieving near-perfect performance for recall and precision. Specifically, the performance on the clinical datasets shows the proposed approach is suitable for practical usage.

Used code, data and results can be found here.

Declaration on Generative AI

Generative AI tools were used in a limited capacity. AI tools were used to increase the visual quality of figures (color selection, fontsize, etc) and minor editing of the text (latex commands, etc), for better readability. The original implementation, experiment design, results, and any conclusions drawn are produced without employing Generative AI tools.

References

[1] M. Gagnon, Ontology-based integration of data sources, in: 2007 10th International Conference on Information Fusion, 2007, pp. 1–8. doi:10.1109/ICIF.2007.4408086.

[2] F. N. AL-Aswadi, H. Y. Chan, K. H. Gan, From ontology to knowledge graph trend: Ontology as foundation layer for knowledge graph, Communications in Computer and Information Science (2022) 330–340. doi:10.1007/978-3-031-21422-6_25.

[3] A. Adya, P. Bahl, J. Padhye, A. Wolman, L. Zhou, A multi-radio unification protocol for ieee 802.11 wireless networks, in: First International Conference on Broadband Networks, 2004, pp. 344–354. doi:10.1109/BROADNETS.2004.8.

[4] L. Dey, Knowledge graph-driven data processing for business intelligence, WIREs Data Mining and Knowledge Discovery 14 (2024). doi:10.1002/widm.1529.

[5] M. De Bonis, F. Minutella, F. Falchi, P. Manghi, A graph neural network approach for evaluating correctness of groups of duplicates, in: International Conference on Theory and Practice of Digital Libraries, Springer, 2023, pp. 207–219.

[6] R. Hoehndorf, M. Dumontier, J. H. Gennari, S. Wimalaratne, B. De Bono, D. L. Cook, G. V. Gkoutos, Integrating systems biology models and biomedical ontologies, BMC systems biology 5 (2011) 124.

[7] P. Manghi, C. Atzori, M. De Bonis, A. Bardi, Entity deduplication in big data graphs for scholarly communication, Data Technologies and Applications 54 (2020) 409–435. doi:10.1108/dta-09-2019-0163.

[8] G. Costa, A. Cuzzocrea, G. Manco, R. Ortale, Data de-duplication: A review, Studies in Computational Intelligence (2011) 385–412. doi:10.1007/978-3-642-22913-8_18.

[9] T. de Groot, J. Raad, S. Schlobach, Analysing large inconsistent knowledge graphs using anti-patterns, Lecture Notes in Computer Science (2021) 40–56. doi:10.1007/978-3-030-77385-4_3.

[10] L. M. Garshol, A. Borge, Hafslund sesam – an archive on semantics, Lecture Notes in Computer Science (2014) 578–592. doi:10.1007/978-3-642-38288-8_39.

[11] A. Sohail, W. u. Qounain, Locality sensitive blocking (lsb): A robust blocking technique for data deduplication, Journal of Information Science 50 (2022) 1400–1413. doi:10.1177/01655515221121963.

[12] X. L. Dong, D. Srivastava, Big data integration, 2013 IEEE 29th International Conference on Data Engineering (ICDE) (2013) 1245–1248. doi:10.1109/icde.2013.6544914.

[13] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: The semantic web: 15th international conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, proceedings 15, Springer, 2018, pp. 593–607.

[14] B. Yang, W. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. URL: http://arxiv.org/abs/1412.6575.

[15] U. Draisbach, F. Naumann, Dude: The duplicate detection toolkit, 2010. URL: https://api.semanticscholar.org/CorpusID:1190647.

[16] Milan, J. Holze, Home, 2022. URL: https://www.dbpedia.org/.

[17] Göttingen University, Mondial database, https://www.dbis.informatik.uni-goettingen.de/Mondial/, 2024. Accessed: 2024-06-10.

[18] E. Huaman, E. Kärle, D. Fensel, Duplication detection in knowledge graphs: Literature and tools, 2020. URL: https://arxiv.org/abs/2004.08257. arXiv:2004.08257.
[19] A. K. McCallum, K. Nigam, J. Rennie, K. Seymore, Automating the construction of internet portals with machine learning, 2000.
[20] S. Mehryar, Resolution-alignment-completion of tabular electronic health records via meta-path generative sampling, in: Proceedings of the 4th Table Representation Learning Workshop, Association for Computational Linguistics, Vienna, Austria, 2025, pp. 200–207. URL: https://aclanthology.org/2025.trl-1.17/.
[21] M. Fey, J. E. Lenssen, Pytorch geometric: Deep learning on graphs and other irregular structures, https://pyg.org/, 2019.
[22] C. Cambier van Nooten, T. van de Poll, T. Heskes, Y. Shapovalova, Gnn deep ensembles for n-1 contingency decisions, in: IET Conference Proceedings CP922, volume 2025, IET, 2025, pp. 2435–2439.
[23] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, Advances in neural information processing systems 30 (2017).