Review of Knowledge Graph Embedding Models for Link Prediction on Wikidata Subsets

Quinton Mays
School of Data Science
University of Virginia
Charlottesville, Virginia
rub9ez@virginia.edu

Antoine Edelman
School of Data Science
University of Virginia
Charlottesville, Virginia
ae4fk@virginia.edu

Olu Omosebi
School of Data Science
University of Virginia
Charlottesville, Virginia
oo7bq@virginia.edu

***Abstract*—Wikidata is a free and open knowledge base that is primarily maintained by a community of volunteers. Due to its scale and open-source nature, many of the items in Wikidata contain missing properties. Knowledge graph embedding (KGE) models provide a method to automatically perform link prediction to aid the Wikidata community in identifying potential data sparse items. While there exist a number of reviews of KGE models for link prediction that utilize similar datasets such as YAGO and Freebase, to our knowledge there are comparatively few reviews that utilize Wikidata as a data source. In this paper we present a review of knowledge graph embedding models implemented in the Python package PyKEEN (Python KnowlEdge EmbeddiNgs)[1] and examine their effectiveness performing link prediction across several different subsets of Wikidata. We also discuss the effectiveness of different hyperparameter configurations on each model. Finally, we make our project's code for efficiently pre-processing Wikidata dumps into small scale subsets, trained knowledge graph embedding models for link prediction, and Wikidata subset datasets available to the Wikidata community at-large for use in future research.**

***Index Terms*—Wikidata, knowledge graph, knowledge graph embeddings, link prediction**

I. Introduction

Extracting knowledge and insight from data is one of the core goals of data science. However, even if effective models exist for given task, many datasets are known to contain inherent flaws. These flaws can stem from a number of sources including instrument calibration and human error. Flaws in the dataset may manifest themselves as bias and error when used to train models. As Wikidata serves as the structured data storage for other Wikimedia Commons projects, including Wikipedia [2], and these datasources are available for use by anyone under the Creative Commons Public Domain Dedication 1.0 license, it is critical that this data is as accurate and representative as possible. Currently, the Wikidata knowledge base is maintained by a community of volunteers who add new items and maintain existing items. As Wikidata is a large knowledge graph with almost 100 million items, the task of completing underserved areas of the knowledge base and evaluating existing data is challenging. Knowledge graph embeddings provide a method to evaluate missing and current links in a knowledge base for their validity. [3] This method, known as link prediction, provides a computational method for aiding the Wikidata community in completing the Wikidata knowledge base.

In this paper, we make two contributions to aid the Wikidata community in creating knowledge graph embeddings for link prediction:

- We present a set of tools for processing Wikidata n-triples dumps from their raw form into subset datasets which can be interpreted by most knowledge graph embedding libraries. These tools are optimized to run on individual computing resources and therefore are accessible to the Wikidata community at large.
- We perform a review of several knowledge graph embedding models' effectiveness at link prediction on several subsets of Wikidata. We also examine the effects of different hyperparameter configurations on the performance of these models.

II. Background

A. Wikidata

"Wikidata is a free, collaborative, multilingual, secondary database, collecting structured data to provide support for Wikipedia, Wikimedia Commons, the other wikis of the Wikimedia movement, and to anyone in the world."[2] Wikidata is published under the Creative Commons Public Domain Dedication 1.0 license, which allows anyone to use the data, in any situation, including for commercial use. It is available in over 100 languages, allowing people from across the world to interact with it using various methods. As the structured data storage for other Wikimedia projects, including Wikipedia, Wikidata is structured for machine readability. As of the time of writing Wikidata contained 98,686,010 items and had been edited 1,675,257,718 times since it's launch in 2012 [4].

Wikidata consists mainly of items and statements [2]. Items consist of a label, which is the item's name, a description of the item, and any aliases for that item. Items are assigned a unique identifier or Q-number. An example of an item would be University of Virginia (Q213439). Statements define item properties and consist of an item, a property, and either a value or another item. An example of an item, property, value statement is University of Virginia (Q213439), students count (P2196), 21000, while one example of an item, property, item

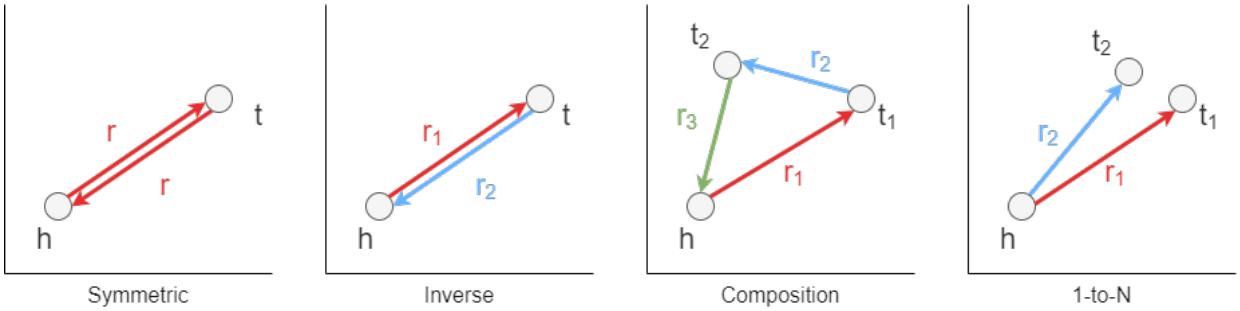


Fig. 1: Common Knowledge Graph Relationship Types

statement is University of Virginia (Q213439), founded by (P112), Thomas Jefferson (Q11812). Statements may also be accompanied by additional qualifiers and references.

Mora-Cantallops et al. [5] state that Wikidata is increasingly popular in research literature, with the majority of research focusing on applications and tools that utilize Wikidata's structure as a knowledge graph. Many of these applications are focused around Natural Language Processing (NLP) and Natural Language Generation (NLG).

B. Knowledge Graphs

Wikidata is an example of a knowledge graph[3, 6]. While there is no widespread formal definition for knowledge graphs[3], we chose to define a knowledge graph as $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{F}\}$, where the graph $\mathcal{G}$ is made up of entities $\mathcal{E}$, relationships $\mathcal{R}$, and facts $\mathcal{F}$. A fact $f \in \mathcal{F}$ is recorded as a triple of the form $f = (h, r, t)$, where $(h, t) \in \mathcal{E}$ are the head and tail entities respectively, and $r \in \mathcal{R}$ is the relationship between h and t.[3]

Each relationship in a knowledge graph has a pattern that defines its interactions with entities. [7, 8, 9] Symmetric relationships are relationships where $r(h, t) \Rightarrow r(t, h) \forall h, t$. Examples of a symmetric relationship are kinship and roommate relationships. Asymmetric relationships, such as the hypernym relation, are those relationships where $r(h, t) \Rightarrow \neg r(t, h) \forall h, t$. In Fig. 1, asymmetric relationships are similar to symmetric relationships but r only goes one direction. Inverse relationships, such as the advisor-advisee relation, are those that satisfy $r_2(h, t) \Rightarrow r_1(h, t)$. Transitive, or composition relations are those relationships that follow the pattern $r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z) \forall x, y, z$. An example of this relationship is if y is x's mother and z is y's husband, then z is x's father. The final relation type is a 1-to-N relation where $r(h, t_1), r(h, t_2), \ldots, r(h, t_n)$ for which all are True. An example of this relation type is the relation "StudentsOf". These relation types are visualized in Fig. 1.

Apart from Wikidata, there exist a number of other open-source general knowledge graphs including WordNet[10], Cyc[11], DBPedia[12], YAGO[13], and Freebase[14] [3]. In addition, many domain specific knowledge graphs such as UMLS[15], SNOMED CT[16], STRING[17] and SKEMPI[18] exist, particularly for use in biomedical applications. Common applications of knowledge graphs are knowledge discovery, question answering, and knowledge graph completion. [19] Knowledge graph completion includes embedding-based ranking, relation path reasoning, rule-based reasoning, and meta relational learning. [3] Link prediction utilizes embedding-based rankings to score triples to predict the plausibility of a given fact.

C. Knowledge Graph Embedding Models

Knowledge graph embedding models create a representation of each fact triple (h, r, t) from a knowledge graph $\mathcal{G}$ as vectors $(\mathbf{h}, \mathbf{r}, \mathbf{t})$ in a representation space using a scoring function. [3] Some common representation spaces are real space $\mathbb{R}$, complex vector space $\mathbb{C}$, and various manifold spaces. Each model implements a scoring function, which measures the plausibility of a given fact. There are two main types of scoring functions, distance based and semantic matching. Distance based functions use a distance measure to measure the distance between $\mathbf{h} + \mathbf{r}$ and $\mathbf{t}$. Semantic scoring functions measure the plausibility of a fact via semantic matching of the entity-relation pairs (h, r) and (r, t).

Knowledge graph completion models can be broadly categorized into one of three groups: translational models, semantic interaction models, and neural network models. [20] Models from each of these categories are implemented in the Python package PyKEEN (Python KnowlEdge EmbeddiNgs)[1], including:

1) Translational Models: **TransE** [21] is a translation embedding model which embeds triples (h, r, t) in $\mathbb{R}^k$, where k is a hyper-parameter for the number of dimensions in which to produce the embedding. The embedding is generated by minimizing the scoring function $f_r(\mathbf{h}, \mathbf{t}) = -||\mathbf{h} + \mathbf{r} - \mathbf{t}||_{1/2}$. Other scoring function norms may also be used as an additional hyper-parameter [9]. TransE assumes that "true" facts will satisfy $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. [20] Due to its structure, TransE cannot model symmetric and 1-to-N relations seen in Fig. 1. [7] **TorusE** [22] is a manifold space variant of the TransE model which embeds relationships on a toroidal surface in the embedding space, in contrast to TransE which produces embeddings on a spherical surface. This change removes some of the limitations of TransE and allows for the modeling of 1-to-N relationships.

2) Semantic Interaction Models: **RESCAL**[23] represents entities and relationships as vectors and matrices. Each re-

lationship is modeled as $\mathbf{W}_r \in \mathbb{R}^{kxk}$ which represents the relationship between the i-th latent factor of $\mathbf{e}_h \in \mathbb{R}^k$ and the j-th latent factor of $\mathbf{e}_t \in \mathbb{R}^k$. k is a hyper-parameter that controls the dimensionality of the embeddings. The plausibility score of each triple is given by $f(\mathbf{h}, \mathbf{r}, \mathbf{t}) = \mathbf{e}_h^T \mathbf{W}_r \mathbf{e}_t$. **DistMult**[24] is a variant of RESCAL that restricts the form of relationship matrices to diagonal matrices. It is unable to model asymmetric, inverse, or composition relations.[7] **ComplEx**[25] is similar to DistMult and uses a diagonal matrix to represent relation embeddings. ComplEx extends DistMult by modeling entities in complex space $\mathbb{C}^k$, where k is a hyper-parameter for the dimensionality of the embedding. Due to this extension, ComplEx is able to differentiate between symmetric and asymmetric relations.[7] It's scoring function is given as $f_r(\mathbf{h}, \mathbf{t}) = Re(\mathbf{e}_h \odot \mathbf{r}_r \odot \bar{\mathbf{e}}_t)$. **PairRE**[26] uses paired vectors to model knowledge graph relationships. It uses the scoring function $f_r(\mathbf{h}, \mathbf{t}) = ||\mathbf{h} \odot \mathbf{r}^H - \mathbf{t} \odot \mathbf{r}^T||$. PairRE has two hyper-parameters γ, which controls the fixed margin for distance based embedding models, and k, the dimensionality of the embedding. PairRE is able to encode symmetry, asymmetry, inverse, and composition patterns, and can capture more complex relations such as 1-N with some constraints. **HolE**[27] uses holographic embeddings to create compositional vector space models of knowledge graphs. It uses the scoring function $f(h, r, t) = \sigma(\mathbf{r}^T(\mathbf{h} \star \mathbf{t}))$ where $\star$ is the circular correlation operator $\mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ defined as $[\mathbf{a} \star \mathbf{b}]_i = \Sigma_{k=0}^{d-1} \mathbf{a}_k * \mathbf{b}_{(i+k) \bmod d}$. HolE is able to model composition relations.

3) Neural Network Models: **ConvE**[28] is a convolutional neural network-based approach for generating knowledge graph embeddings. It accepts a matrix $A \in \mathbb{R}^{2xn}$ as an input, where the first row contains the head h and the second row contains the relationship r. It then reshapes this matrix into an "image" matrix $B \in \mathbb{R}^{mxn}$ and applies a set of 2-dimensional convolutional filters to B. The output of the convolutional layer is then projected to the chosen embedding dimension and multiplied with the entity matrix $\mathcal{E}$ to produce logits. The logisitc sigmoid function is then applied to the logits to produce output scores. It is efficient at computing scores for 1-N relationship, where the head and tail are fixed. **ConvKB**[29] also employs a convolutional neural network to model knowledge graph data. It accepts an input matrix $A = [\mathbf{h}; \mathbf{r}; \mathbf{t}] \in \mathbb{R}^{nx3}$ and applies τ 1x3 convolutional filters, where τ is a hyper-parameter. The results of the convolutional layer are then concatenated to each other and then the dot product is taken with a weights matrix $\mathbf{w} \in \mathbb{R}^{\tau n X 1}$ to determine the score for each triple.

D. Model Training Considerations

In addition to the choice of model and its associated hyper-parameters, there are several other considerations for creating knowledge graph embeddings. As most knowledge graphs contain only positive training examples, training approaches that introduce negative sampling to avoid over-generalization on true facts must be introduced.[9] Two approaches exist: local closed world assumption (LCWA)[30] and stochastic locally closed world assumption (sLWCA). Under the locally closed world assumption, a set of negative examples $\mathcal{H}^-(r, t)$, $\mathcal{R}^-(h, t)$, or $\mathcal{T}^-(h, r)$ containing all triples $(h_i, r, t) \notin \mathcal{G}$, $(h, r_i, t) \notin \mathcal{G}$, or $(h, r_i, t) \notin \mathcal{G}$ respectively is generated, and triples in this set are assumed to be false. Under the stochastic locally closed world assumption, instead of considering all triples in $\mathcal{H}^-(h, r)$, $\mathcal{R}^-(h, r)$, or $\mathcal{T}^-(h, r)$ as false, we randomly take samples from these sets. Two possible sampling strategies exist for the sLWCA assumption[9]: uniform negative sampling[21] and Bernoulli negative sampling[31]. Importantly, both the LCWA and sLCWA assume that triples not present in the knowledge graph are False, hence the locally closed world assumption. This assumption does not hold true for all knowledge graphs, as triples not in the knowledge graph may be True.[9].

Another consideration is the choice of loss function[9]. Loss functions can come from one of three categories: pointwise, pairwise, and setwise. Pointwise loss functions compute an independent loss term for each triple-label pair[9]. Pointwise loss functions include Square Error Loss, Binary cross entropy loss, Pointwise Softplus loss, and Pointwise Hinge Loss. Pairwise loss functions compare the scores of positive triples and negative triples obtained by corrupting the positive triple. Examples of Pairwise functions include Margin ranking loss and Pairwise logistic loss. Setwise loss functions consider the combined score of more than two triples. Examples of Setwise loss functions are Self-adversarial negative sampling loss and Cross entropy loss.

A third consideration for training is the choice to explicitly create embeddings for inverse relations[9]. An inverse triple (h, r_{inv}, t) is a relation that reverses the relationship r in the triple (h, r, t). To do so effectively extends the number of training examples and alters the scoring function such that predicting the head h for (r, t) becomes predicting tail entities for (t, r_{inv}).

E. Model Evaluation

Knowledge graph embedding models are evaluated based on their performance on the link prediction task, or how well they predict the head/tail entities for (r, t) and (h, r) respectively.[9] To assess this performance, negative examples must be generated, as true negative examples may not be available. These negative examples are generated from the test by creating two sets $\mathcal{H}(r, t) = \{(h', r, t) | h' \in \mathcal{E} - \{h\}\}$ and $\mathcal{T}(h, r) = \{(h, r, t') | h \in \mathcal{E} - \{t\}\}$ containing all triples where the head and tail entities are corrupted. However, these generated negative examples may contain true triples, which may distort the results. To alleviate this issue Ali et al. [9] presents a filtered approach where known triples are excluded from $\mathcal{T}(h, r)$ and $\mathcal{H}(r, t)$[21].

Once the corrupted triples are generated and filtered, a variety of metrics may be applied to assess the generated embeddings. **Mean rank** calculates the average scoring rank of the triples in the test set:

$$MR = \frac{1}{|\mathcal{G}_{test}|} \sum_{t \in \mathcal{G}_{test}} rank(t)$$

Smaller values of mean rank indicate a higher performing model. Mean rank is reliant on the size of the test set $\mathcal{G}_{test}$, making comparisons between different size graphs difficult. To remedy this, **Adjusted mean rank (AMR)**[9][32] compares the model to a model with random scores. AMR is given by:

$$AMR = \frac{MR}{\frac{1}{2}\sum_{t \in \mathcal{G}_{test}}(\xi(t)+1)}$$

where ξ is the number of triples which the true triple t is compared against. AMR measures fall in the range $[0, 2]$, where smaller values indicate a higher performing model. **Hits@K** is another measure of model performance that measures the number of true facts ranked in the top K scoring triples.

$$Hits@k = \frac{|\{t \in \mathcal{G}_{test} | rank(t) \leq k\}|}{|\mathcal{G}_{test}|}$$

Common values for k are 1, 3, 5, and 10.

III. Methods

A. Wikidata Subsets

To begin our analysis, we downloaded the most recent n-triples (.nt) dump from Wikidata. This file was .bz2 compressed and was approximately 30GB in size. As the decompressed file had only a portion of the needed data for this project, we determined that file must be incrementally decompressed and processed. The n-triples format stores three main types of triples as lines in the file.

- *Entity-Label-Value (ELV)*, triples contain metadata for each Wikidata item, such as item names and aliases in multiple languages.
- *Entity-Property-Value (EPV)*, triples contain valued properties for each Item, such as population, GDP, phone numbers, and other non-entity values.
- *Entity-Property-Entity (EPE)*, triples contain the relationship triples in the form (h, r, t), and are used to build the knowledge graph embeddings.

To process the file, we utilize Python's built-in multiprocessing and bz2 libraries to break the file into chunks which are passed to workers to parse. The parsing operation uses regular expressions to remove unnecessary hyperlink text and to determine a triple type (ELV, EPV, or EPE), then writes the processed triple to a respective buffer. Once each buffer is full, the main thread then writes it to a tab-separated values (.tsv) file. The output of this processing is three .tsv files, one containing the EPE triples, one containing the EPV triples, and one containing the ELV triples for use as metadata. Only the EPE triples file is used in this work. By utilizing multiprocessing, this process reduces the processing time on a machine with an 8th generation Intel i7 CPU and 16GB of RAM from over 10 hours when processed on a single worker to under 2 hours when processed with 8 cores.

To create subsets for analysis, our team created a Python script that utilizes the Dask library[33] to parallelize filtering of the processed EPE .tsv file. The script first queries Wikidata's SPARQL endpoint to retrieve a list $\mathcal{C}$ of all subclasses of a user defined class. This list is then used to create a filter for the Dask dataframe $\mathcal{I} = (h, r_{instance}, c) \forall c \in \mathcal{C}$, where $r_{instance}$ is a special property, P31, indicating the head entity h is an instance of class c. The instance list $\mathcal{I}$ is then used to create the subset $\mathcal{S} = (h, r, t) \forall (h, t) \in \mathcal{I}$. The decision was made to restrict $\mathcal{S}$ to only triples where $(h, t) \in \mathcal{I}$ to reduce the size of each subset and to inform future research on domain specific knowledge graphs related to these subsets. To further reduce the size of some subsets, additional filters were applied. For the humans dataset, a filter for nationality was applied to limit the items to those with property P27 "Country of Citizenship", Q30 "United States of America". The incorporated places dataset, which was originally intended as a corporations subset, was created using the Python library NetworkX [34] as an induced subgraph of all neighbors within a radius four of a random node in a Corporations subset. In this case, the randomly chosen item was Tours (Q288), a city and commune in Indre-et-Loire, Centre-Val de Loire, France, causing the subgraph to contain only incorporated places, and no business corporations. In addition to its use in creating subsets, NetworkX was also used to generate graph topology measures for each of the subsets and to generate visualizations of each dataset as seen in Fig. 2. This visualization shows the nodes in the graphs with a high degree of edges colored in red. Most of the nodes in the graphs have lower degrees and they are displayed in blue. The different colors makes it easier to visualize the structure of the graph and to easily identify the important nodes in the graph structure. An overview of each subset may be found in Table. I.

B. Model Training and Evaluation

To train and evaluate models, we utilized the University of Virginia's Rivanna high-powered computing cluster. Rivanna provides access to up to 28 CPU cores and 4 GPUs for training models. As PyKEEN operates using Pytorch [35], GPUs can be utilized for efficient model training. All models were trained using a single CPU with 6GB of memory and an NVIDIA Tesla V100 GPU.

Hyper-parameter tuning was conducted using PyKEEN's hyper-parameter optimization pipeline. This pipeline utilizes a Tree-structured Parzen Estimator (TPE)[36] to determine optimal model hyper-parameters given a search space and a fixed number of trials. Each model was trained using the sLCWA assumption and a softplus loss function. This choice was made based on Ali et al. [9]'s large-scale benchmarking of PyKEEN models. Model optimization was conducted using the Adam optimizer[37]. To reduce compute times, early stopping was utilized with a frequency of 50 epochs and a patience of 100 epochs. For all other hyper-parameters, we utilized the default hyper-parameter search spaces from PyKEEN, which were taken from the large-scale benchmarking conducted by Ali et al.[9]. Twenty-five trials were conducted for each model on each dataset.

IV. Results

The results of the twenty-five trials for each model on each dataset are displayed in Fig. 3 for Hits@10 and Fig 4 for AMR.

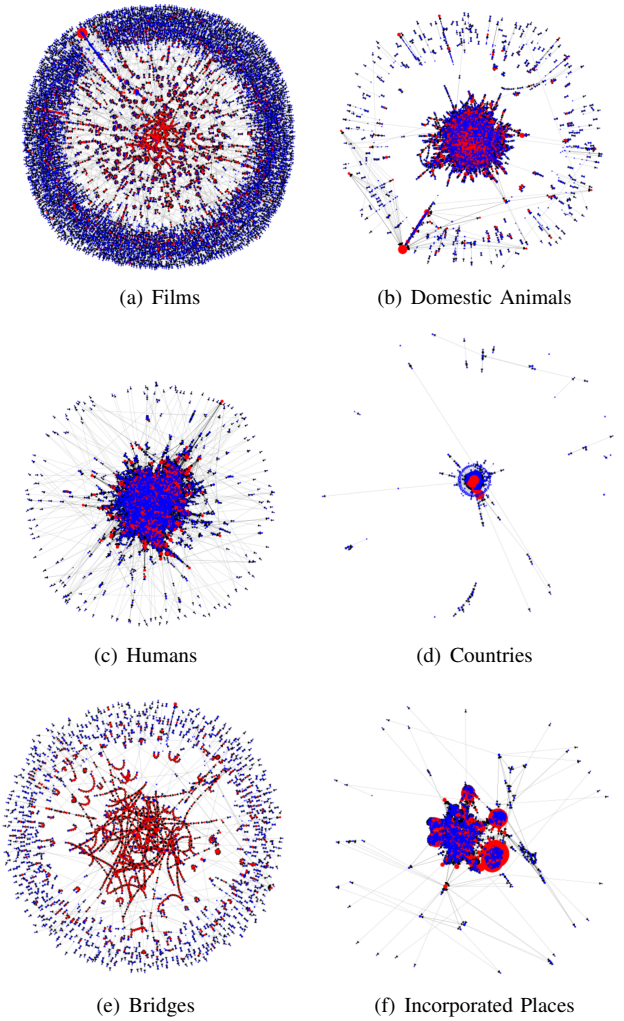
Dataset	Triples	Unique Entities	Unique Relations	Avg. Degree	Density	Example Triple *(h,r,t)*
Bridges	4,472	2,736	32	1.81	0.0005	*(Milburngate Bridge, next crossing upstream, Framwellgate Bridge)*
Incorporated Places	30,875	4,408	27	32.57	0.0015	*(Boston, shares border with, Cambridge)*
Countries	7,933	731	29	75.73	0.04	*(Belgium, diplomatic relation, Luxembourg)*
Domestic Animals	21,000	8,765	25	5.36	0.0002	*(Baroquer, child, Bayan Dama)*
Films	12,454	9,512	55	2.81	0.0001	*(Crazy Romance, different from, Crazy Romance)*
Humans	42,621	7,402	35	13.31	0.0008	*(Louis XIV, father, Louis XIII)*

TABLE I: Wikidata Subsets

Note that higher Hits@10 results indicate a better performing mode, while lower AMR scores indicate higher performing models. A summary of the top three models on each subset may be found in Table IV.

Model	Symmetric	Asymmetric	Inverse	Compositon.	1-to-N
ComplEx	✓	✓	✓	X	✓
ConvE	?	✓	✓	?	✓
ConvKB	X	✓	✓	✓	✓
DistMult	✓	X	X	X	✓
HolE	✓	✓	✓	✓	✓
PairRE	✓	✓	✓	✓	✓
RESCAL	✓	✓	✓	✓	?
TorusE	?	✓	✓	✓	✓
TransE	X	✓	✓	✓	X

TABLE II: Model Ability to Represent Relations



(a) Films (b) Domestic Animals (c) Humans (d) Countries (e) Bridges (f) Incorporated Places

Fig. 2: Subset Topology Visualization

A. Bridges

The bridges dataset is a subset of the Wikidata knowledge graph and consists of the bridges' relationships. It is a sparse graph with a density of 0.0005, 2,979 nodes, and 4,473 edges. The bridge with the most number of relationships is the Rotterdam bridge (Q3980145) in the Netherlands, with 39 edges. Most of the bridges in the dataset have one to four edges. Some examples of relationships between Bridges in the dataset are P1889 (different from), P1365(replaces), P177(crosses), and P138(named after). Complex with an embedding dimension of k = 192 is the best performing model, and from Fig. 3, the k value of 192 provides the best AMR results for all models in this dataset. In Fig. 1, Complex outperforms the other models with the highest Hits@10 result of 0.86, and it also outperforms other models with its low AMR score of 0.09. Other competitive models with good results in Fig. 1 and 2 are ConvKB and RESCAL, as indicated by their interquartile ranges. Complex can capture asymmetric relationships in triples, and 48.7 percent of the bridges' triples are asymmetric, and the graph topology of the Bridges subset is in Fig 5(e).

B. Incorporated Places

The incorporated places subset contains the relationships between over 4,000 unique places. It is an interesting dataset due to its construction from the larger corporations dataset and represents a smaller location granularity than the countries dataset. Fig. 2(f) provides a visualization of this graph, which contains several densely connected regions surrounded by sparsely connected regions. ComplEx is the best performing model overall on this subset, with the best Hits@10 score of 0.830 and the second best adjusted mean rank score of 36.02. This model had an embedding dimension of $k = 32$, which is one of the smaller k values in the search space. This may indicate that a majority of the information in the dataset can be captured in a relatively small number of dimensions using ComplEx. Fig. 5 displays AMR performance variance

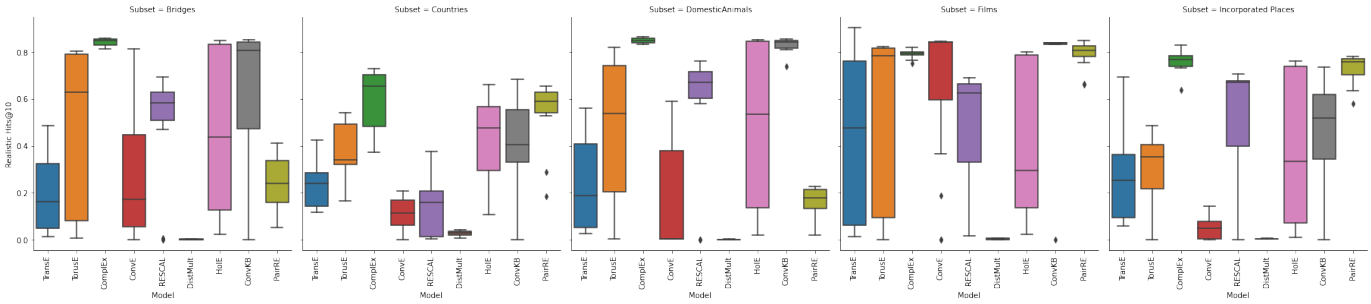


Fig. 3: Distribution of Hits@10 Performance by Subset and Model

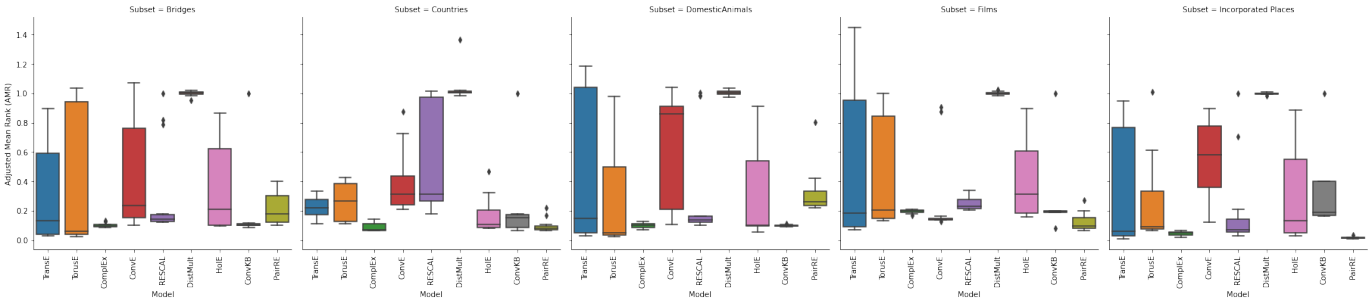


Fig. 4: Distribution of AMR Performance by Subset and Model

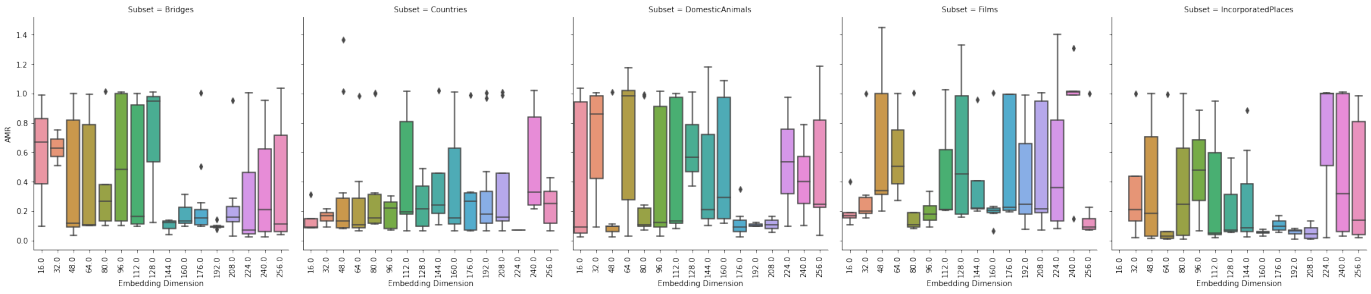


Fig. 5: Distribution of AMR Performance by Model Embedding Dimension (k)

by embedding dimensions and indicates that for $k \in [116, 204]$ the AMR variance is smaller and may represent a better choice across all models. A summary of the top-3 performing models and associated hyper-parameters may be found in Table III. Interestingly, TransE has the best model overall when evaluated by mean rank and mean reciprocal rank. However, TransE has a much higher variance in MR and AMR as seen in Fig. 4. The example triple for this dataset, *(Boston, shares border with, Cambridge)*, is an example of a symmetric relationship. This relationship, P47, is part of more than 50% of all triples in the dataset. The prevalence of this relationship may explain the high performance variance of TransE, as both TransE and ComplEx can model asymmetric and inverse relations as shown in Table II, while only ComplEx can model symmetric relations. Fig. 6 displays the training times for each model and indicates that ComplEx was also one of quickest training models as demonstrated in Fig .6.

C. Countries

The countries dataset is one of the smaller datasets as seen in Table I, with under 8,000 triples and 29 relation types. It is similar to the Nations dataset from Boschin [38]'s wikidatasets Python package, a common Wikidata KGE dataset. Fig. 2(d) demonstrates that this graph is characterized by a densely connected core with sparsely connected outlier nodes. As seen in the Table I example triple, this dataset is dominated by triples containing relation *P530 "Diplomatic Relation"*, with 6,513, or more than 80%, of triples containing this relationship. This relationship is an example of a symmetric relationship, where both (h, r, t) and (t, r, h) are true. Fig. IV and Table III demonstrate that the best performing model for this subset is ComplEx, followed by PairRE. These models are able to encode symmetric relations as seen in Table II, which explains their high performance on this dataset. Intriguingly DistMult, another model that is able to encode symmetric relationships, does not perform well on this dataset. As this model performs

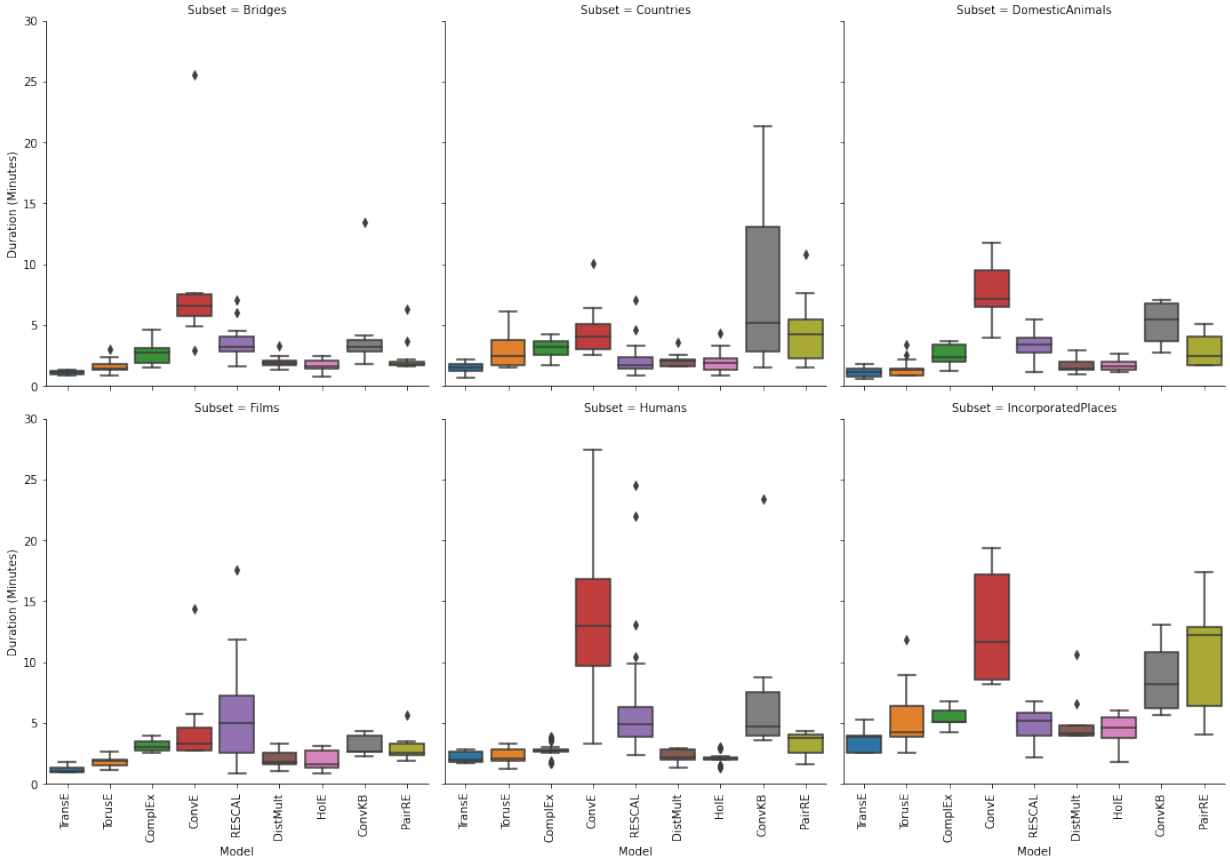


Fig. 6: Distribution of Training Time by Subset and Model

poorly on many of the subset datasets, this may be due to the chosen hyper-parameter search space, as only smaller embedding dimensions were explored due to computational constraints. For this subset, ComplEx performed best when the embedding dimension $k >= 128$, despite other models performing well at lower values of k. For this subset, the best performing model, ComplEx, has the fourth slowest training time, with TransE being the fastest, and ConvKB having the slowest time as shown in Fig .6.

Yo

D. Domestic Animals

The Domestic Animals dataset is a subset of the Wikidata knowledge graph that consists of tamed animals that live with humans. It is a sparse graph with a density of 0.002, 8,855 nodes, and 21,000 edges. There are 25 unique relationships in the dataset, and some examples are P22(father), P40(child), P25(mother), P4743(animal breed), and P1038(relative). The animal with the most number of relationships is Sunday Silence(Q180387), a thoroughbred racehorse with 246 edges. Most of the other animals in the dataset have 1 to 8 edges, and there are 21,000 triples in this dataset, and 47.8 percent of these triples are asymmetric. From Fig. 1, Complex is the best performing model with a hits@10 result of 0.86, and ConvKB is close in performance. Rescal and Dismult have the lowest performance. Although Complex and Dismult are both bilinear models, Dismult cannot recognize asymmetric triples. Table II indicates that Complex has an AMR score of 0.86 and a k value of 32. Fig.3 shows that the k value with the best results for embedding this dataset is a k value of 48, and a graphical illustration of the Domestic Animals dataset is shown in fig 5(b).

E. Films

The Films dataset is a subset of Wikidata that consist of cinematic work. It is a sparse graph with 12,454 unique triples, a density of 0.0001, 11,275 nodes, and 12,454 edges, and there are 55 unique relationships in the dataset. The node with the highest number of relationships is the movie Women Make Film (Q77855044), with 303 edges. Most of the other nodes in the dataset have between 1 and 6 edges. Some examples of relationships in the dataset are P4585(first appearance), P1441(present in work), and P180(depicts). From Fig. 1, TransE has the best Hits@10 result of 0.90, although Complex has the overall best performance. There are 11,275 nodes in the Film dataset. 70 percent of the nodes have two or more edges, and 21 percent have three or more edges. TransE works well on this kind of graph structure. In Fig. 2 TransE achieved the best AMR score of 0.07, and it also has the largest variance of AMR score results. TransE also has the fastest time

Subset	Rank	Model	k	Norm	γ	Hits@10	AMR	MR
Films	1	TransE	256	2	-	0.90	0.07	402
	2	TransE	208	2	-	0.89	0.07	403
	3	TransE	256	1	-	0.89	0.07	400
Domestic Animals	1	ComplEx	32	-	-	0.86	0.09	400
	2	ComplEx	16	-	-	0.86	0.08	363
	3	ComplEx	240	-	-	0.86	0.10	456
Bridges	1	ComplEx	192	-	-	0.86	0.09	405
	2	ComplEx	208	-	-	0.86	0.09	412
	3	ComplEx	192	-	-	0.86	0.09	385
Incorporated Places	1	ComplEx	32	-	-	0.83	0.02	36
	2	ComplEx	240	-	-	0.79	0.03	60
	3	PairRE	64	-	1	0.78	0.01	21
Countries	1	ComplEx	160	-	-	0.73	0.06	12
	2	ComplEx	256	-	-	0.71	0.06	12
	3	ComplEx	128	-	-	0.70	0.06	13
Humans	1	TorusE	160	-	1	0.00	1.01	3720
	2	TorusE	160	-	1	0.00	1.01	3720
	3	TorusE	160	-	1	0.00	1.01	3720

TABLE III: Top-3 Models and Hyperparameters by Subset

among the trained models, and a graphical representation of the Films dataset is shown in fig 5(a).

F. Humans

The humans subset contains people connected to Louis XIV (Q7742) within six degrees of separation. Its relationships are only of that between humans (Q5). An example triple can be seen in Table I. While it is commonly connected to other data sets like countries (since many in Wikidata are rulers or political officials), it does not hold those characteristics. This subset performed the worst of the subsets, so low that the results are not displayed in the Fig. 3, 4, and 5. Table IV shows that ComplEx (AMR = 0.971, MR = 3592) and TorusE (Hits@10 = 0.003) performed the best on humans. Yet, they are not significant enough to consider for production. What is odd is it's ratio of unique entities to triples is closest to Countries (the smallest subset). Also, the average degree and density rather middle of the back. Meaning that basic topology hardly differentiates humans from the others. However, it has 42,622 triples which is the largest of all the datasets by a large margin.

V. Discussion

The results shown in Table IV indicate that ComplEx is consistently among the top performing models across all subsets when evaluated using the metric Hits@10. This strong performance is consistent with the benchmarking results in the literature review, including Ali et al. [9]. When evaluating using this metric, PairRE also is a strong candidate across all subsets, particularly the Countries subset. This may be due to the higher average degree of the countries subset compared to other subsets. DistMult is among the worst candidate models across all datasets, with poor Hits@10 and AMR scores. This result is surprising, as the structure of DistMult is closely related to ComplEx. However, DistMult cannot model asymmetric or inverse relations, which may explain its poor performance on all these datasets.

Because ComplEx performs best at Hits@10, it indicates that it may be strong at certain predictions. This means that TorusE and TransE are doing better over all but not getting as strong rankings as ComplEx. Since ComplEx is unable to understand composition relationships, this is were TransE has the advantage by being able to make predictions [9]. However, symmetric and 1-to-N relations are not captured by TransE. The different relationships that the model can predict effect the metric results. Moving forward, the following models seem to have a ability to make predictions on these relationship types: TransH, TransR, and TransD [9]. However, changing the model type is not necessarily what is needed to get better results.

The largest subset, humans, did not produce any quality results as seen in Table IV compared to the other subsets. This may be due to its different structure as seen in Fig. 2 compared to other subsets. More noticeably the triples being several magnitudes larger. However, training times from Fig. 6 show that models seem to run similar to humans especially Incorporated Places. Yet, due to the size difference and results, the early stopping is what caused the timing to look similar. Because of the size of Humans, more epochs without triggering early stopping is suggested with larger hyper-parameter ranges to gather meaningful results.

There is a correlation between the average degree and max performance after 100 trials. However, based on the ranges of performance from Fig. 3 & Fig. 4, more trials will get a better distribution of results. It seems that only ComplEx reached near optimal performance in the hyper-parameter pipeline. More trial are recommended to ensure that gradient-decent has reached its lowest point. Paired with a more sophisticated loss optimiser than Adam (although good on its own) should be seeing that result at 200 trials.

The strong performance on several of the subset models may indicate that it may be advantageous to create an ensemble of many small subset KGE models rather than creating a single large model for large knowledge bases. This approach reduces the needed computational resources and could reduce training time. However, using an ensemble of subset models would not allow for intra-subset connections to be predicted.

		Subset					
Model	*Metric*	Bridges	Incorporated Places	Countries	Domestic Animals	Films	Humans
ComplEx	AMR	0.087	0.016	0.063	0.072	0.170	**0.971**
	MR	384.8	36.02	12.33	317.8	957	**3592**
	Hits@10	**0.860**	**0.830**	**0.730**	**0.864**	0.820	0.002
ConvE	AMR	0.102	0.119	0.212	0.109	0.125	0.986
	MR	449.4	262.1	41.27	481.4	704.6	3648
	Hits@10	0.812	0.141	0.206	0.590	0.844	0.002
ConvKB	AMR	0.087	0.165	0.065	0.091	0.083	0.992
	MR	383.7	362.4	12.67	401.6	466.8	3672
	Hits@10	0.853	0.734	0.683	0.854	0.839	0.003
DistMult	AMR	0.954	0.984	0.982	0.971	0.984	0.982
	MR	4222	2158	190.9	4298	5546	3633
	Hits@10	0.003	0.006	0.042	0.003	0.008	0.002
HolE	AMR	0.094	0.027	0.080	0.056	0.156	0.993
	MR	417.5	58.33	15.64	249.2	881.2	3674
	Hits@10	0.847	0.760	0.660	0.853	0.801	0.002
PairRE	AMR	0.099	0.010	**0.063**	0.222	**0.064**	0.988
	MR	439.5	21.38	**12.27**	981.5	**363.2**	3657
	Hits@10	0.413	0.781	0.654	0.227	0.848	0.003
RESCAL	AMR	0.120	0.028	0.177	0.100	0.205	0.984
	MR	530.8	61.15	34.51	442.8	1158	3642
	Hits@10	0.692	0.705	0.375	0.760	0.689	0.003
TorusE	AMR	**0.023**	0.066	0.112	**0.022**	0.134	0.992
	MR	**100.3**	144.1	21.77	**99.01**	755.5	3669
	Hits@10	0.802	0.485	0.540	0.821	0.822	**0.003**
TransE	AMR	0.027	**0.008**	0.112	0.031	0.071	0.986
	MR	118.8	**17.72**	21.85	139.3	399.9	3649
	Hits@10	0.486	0.694	0.423	0.560	**0.905**	0.002

TABLE IV: Best Model Results

Further research is needed in this area to compare these two approaches.

VI. Conclusions and Future Work

In this paper we presented a review of the effectiveness of several knowledge graph embedding models on subsets of Wikidata. Our analysis determined that the ComplEx is an excellent candidate model for a wide range of Wikidata subsets. This finding is aligned with many of the benchmarks conducted on other subsets in the reviewed literature, indicating that these findings' validity may extend to Wikidata. Our analysis also determined that the proportions of each type of relation in the knowledge graph has a distinct effect on the performance of each model architecture. However our analysis of relationship types was limited to manual examination of the data. Future research could focus on creating ensembles of predictors based on automatic identification of relationship types. Finally, we analyzed the effectiveness of different hyper-parameter configurations on each subset.

We also present a set of tools for pre-processing Wikidata dumps for use in creating knowledge graph embeddings. These tools reduce the time and space required for processing Wikidata dumps. This code is available for use under the MIT license at https://github.com/q-maze/wikidatatools. There we also make available the subset datasets examined in this paper. We hope that these tools are of use to the Wikidata community.

Due to the large scale of Wikidata, our work focused on small scale subsets. For future work, we would like to explore the effectiveness of the aforementioned models at producing embeddings of larger subsets of Wikidata and on the entirety of Wikidata. This could be accomplished using PyKeen's Pytorch Lightning integration, which is relatively new at the time of this writing. This framework provides the ability to parallelize GPU training for larger models. This can potentially be the solution for subsets like humans. In addition to larger subsets, larger hyper-parameter search spaces with additional trial models may yield improved results. We tested two expanded searches on the Nations dataset, which yielded better results. Finally, an exploration of the effectiveness of individual subset embeddings may be compared against larger embeddings, including embeddings of the entirety of Wikidata, such as the one generated by the PyTorch Big Graph team. [39] Regardless of what is found to be the best method of making link predictions in Wikidata, the results in this paper should guide the community in their methodology of finding an optimal solution.

Acknowledgement

Our team would like to thank Will Kent, Wikidata Program Manager at WikiEducation and Dr. Yuri Malitsky for their excellent support and guidance throughout this project.

References

[1] M. Ali, M. Berrendorf, C. T. Hoyt, L. Vermue, S. Sharifzadeh, V. Tresp, and J. Lehmann, "PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings," *Journal of Machine Learning Research*, vol. 22, no. 82, pp. 1–6, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-825.html

[2] "Wikidata introduction," 2022. [Online]. Available: https://www.wikidata.org/wiki/Wikidata:Introduction

[3] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[4] "Wikidata statistics," 2022. [Online]. Available: https://www.wikidata.org/wiki/Wikidata:Statistics

[5] M. Mora-Cantallops, S. Sánchez-Alonso, and E. García-Barriocanal, "A systematic literature review on wikidata," 2019.

[6] A. Hogan, E. Blomqvist, M. Cochez, C. D'Amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A. C. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann, "Knowledge graphs," *ACM Computing Surveys*, vol. 54, 2021.

[7] J. Leskovec, "Cs224w: Machine learning with graphs - kg completion with embeddings," February 2021. [Online]. Available: http://web.stanford.edu/class/cs224w/

[8] Z. Sun, Z. H. Deng, J. Y. Nie, and J. Tang, "Rotate: Knowledge graph embedding by relational rotation in complex space," 2019.

[9] M. Ali, M. Berrendorf, C. T. Hoyt, L. Vermue, M. Galkin, S. Sharifzadeh, A. Fischer, V. Tresp, and J. Lehmann, "Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[10] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[11] D. B. Lenat, "Cyc: A large-scale investment in knowledge infrastructure," *Commun. ACM*, vol. 38, no. 11, p. 33–38, nov 1995. [Online]. Available: https://doi.org/10.1145/219717.219745

[12] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference*, ser. ISWC'07/ASWC'07. Berlin, Heidelberg: Springer-Verlag, 2007, p. 722–735.

[13] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A large ontology from wikipedia and wordnet," *Journal of Web Semantics*, vol. 6, no. 3, pp. 203 – 217, 2008, world Wide Web Conference 2007Semantic Web Track.

[14] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 1247–1250. [Online]. Available: https://doi.org/10.1145/1376616.1376746

[15] O. Bodenreider, "The unified medical language system (umls): integrating biomedical terminology," *Nucleic acids research*, vol. 32, no. suppl_1, pp. D267–D270, 2004.

[16] K. Donnelly *et al.*, "Snomed-ct: The advanced terminology and coding system for ehealth," *Studies in health technology and informatics*, vol. 121, p. 279, 2006.

[17] D. Szklarczyk, A. Franceschini, S. Wyder, K. Forslund, D. Heller, J. Huerta-Cepas, M. Simonovic, A. Roth, A. Santos, K. P. Tsafou *et al.*, "String v10: protein–protein interaction networks, integrated over the tree of life," *Nucleic acids research*, vol. 43, no. D1, pp. D447–D452, 2015.

[18] J. JankauskaitÄ—, B. JimÃ©nez-GarcÃa, J. DapkÅnas, J. FernÃ¡ndez-Recio, and I. H. Moal, "SKEMPI 2.0: an updated benchmark of changes in proteinâ€“protein binding energy, kinetics and thermodynamics upon mutation," *Bioinformatics*, vol. 35, no. 3, pp. 462–469, 07 2018. [Online]. Available: https://doi.org/10.1093/bioinformatics/bty635

[19] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

[20] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan, "Knowledge graph completion: A review," 2020.

[21] A. Bordes, N. Usunier, A. Garcia-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," 2013.

[22] T. Ebisu and R. Ichise, "Toruse: Knowledge graph embedding on a lie group," 2018.

[23] M. Nickel, V. Tresp, and H. P. Kriegel, "A three-way model for collective learning on multi-relational data," 2011.

[24] B. Yang, W. tau Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," 2015.

[25] T. Trouillon, J. Welbl, S. Riedel, E. Ciaussier, and G. Bouchard, "Complex embeddings for simple link prediction," vol. 5, 2016.

[26] L. Chao, J. He, T. Wang, and W. Chu, "Pairre: Knowledge graph embeddings via paired relation vectors," 2021.

[27] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," 2016.

[28] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2d knowledge graph embeddings," 2018.

[29] D. Q. Nguyen, T. D. Nguyen, D. Q. Nguyen, and D. Phung, "A novel embedding model for knowledge base completion based on convolutional neural network," vol. 2, 2018.

[30] X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 -*

27, 2014, 2014, pp. 601–610, evgeniy Gabrilovich Wilko Horn Ni Lao Kevin Murphy Thomas Strohmann Shaohua Sun Wei Zhang Geremy Heitz. [Online]. Available: http://www.cs.cmu.edu/ nlao/publication/2014.kdd.pdf

[31] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, Jun. 2014. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/8870

[32] M. Berrendorf, E. Faerman, L. Vermue, and V. Tresp, "Interpretable and fair comparison of link prediction or entity alignment methods with adjusted mean rank," 02 2020.

[33] Dask Development Team, *Dask: Library for dynamic task scheduling*, 2016. [Online]. Available: https://dask.org

[34] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," 1 2008. [Online]. Available: https://www.osti.gov/biblio/960616

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[36] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in neural information processing systems*, vol. 24, 2011.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[38] A. Boschin, "Wikidatasets : Standardized sub-graphs from wikidata," *CoRR*, vol. abs/1906.04536, 2019. [Online]. Available: http://arxiv.org/abs/1906.04536

[39] A. Lerer, L. Wu, J. Shen, T. Lacroix, L. Wehrstedt, A. Bose, and A. Peysakhovich, "PyTorch-BigGraph: A Large-scale Graph Embedding System," in *Proceedings of the 2nd SysML Conference*, Palo Alto, CA, USA, 2019.