

Explaining Link Prediction with Kelpie

Andrea Rossi¹, Donatella Firmani², Paolo Merialdo¹ and Tommaso Teofili¹

¹Roma Tre University, Rome, Italy

²Sapienza University, Rome, Italy

Abstract

Link Prediction (LP) is the problem of inferring new facts in a Knowledge Graph from the already known ones. Recent advances in Machine Learning have led researchers to develop LP models that represent Knowledge Graph elements as vectors in an embedding space. This approach has led to greatly encouraging results, often outperforming traditional approaches; its main shortcoming is its opaqueness, hindering both our understanding and our trust in these models. In this context, we discuss the Kelpie explainability framework. Kelpie can be applied to any embedding-based LP model and can identify the combinations of training facts that have enabled the prediction of a given link. Kelpie can extract two complementary types of explanations, that we dub *necessary* and *sufficient*.

Keywords

Knowledge Graph Embeddings, Link Prediction, Explainable AI

1. Introduction

Knowledge Graphs (KGs) provide a natural way to model structured real-world information. The nodes of a KG represent real-world *entities*, and they are connected by directed edges denoted by labels conveying semantic *relations*. Each edge connects two entities, forming a triple that is called a *fact*. In time, several KGs have achieved web-scale size, e.g., DBpedia, Yago, Wikidata, and, in industry, the Google KG; nonetheless, all KGs suffer from *incompleteness*, as they miss large portions of the information they should contain. For instance, it has been observed that, in the FreeBase KG, over 70% of the person entities have unknown birthplace, and over 99% have unknown ethnicity [1]. Link Prediction (LP) aims at inferring new, missing facts by analyzing those already present in the KG. For instance, in the graph in Figure 1 (left), knowing $\langle \text{Barack_Obama}, \text{born_in}, \text{Honolulu} \rangle$ and $\langle \text{Honolulu}, \text{located_in}, \text{USA} \rangle$, we can probably infer $\langle \text{Barack_Obama}, \text{nationality}, \text{USA} \rangle$. In the last decade, LP researchers have achieved extremely promising results with the use of *KG embeddings*, i.e., numerical vectors mapped to each element in the KG, and learned with Machine Learning (ML) techniques leveraging the known facts. KG Embeddings have rapidly become the dominant approach to LP in literature, with dozens of new models proposed every year (see [2] for a survey).


Like most ML systems, LP models based on KG Embeddings are inherently opaque, as they do not provide any insights on the reasons *why* they predict specific links. We argue that interpretability in LP systems is vital. Explanations reveal which patterns our models

SEBD 2022: The 30th Italian Symposium on Advanced Database Systems, June 19-22, 2022, Tirrenia (PI), Italy

✉ andrea.rossi3@uniroma3.it (A. Rossi); donatella.firmani@uniroma1.it (D. Firmani); paolo.merialdo@uniroma3.it (P. Merialdo); tommaso.teofili@uniroma3.it (T. Teofili)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

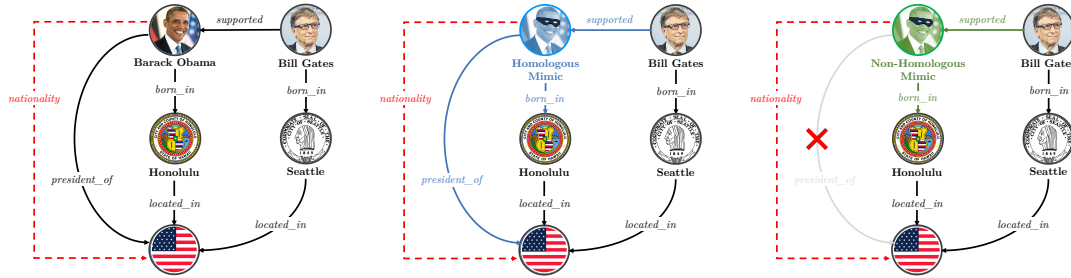


Figure 1: A small KG example (left), together with an example of homologous (center) and non-homologous (right) mimic. The missing fact $\langle \text{Barack_Obama}, \text{nationality}, \text{USA} \rangle$ is denoted in red.

are leveraging to perform their predictions: in the LP field, this would provide a pathway to gain a deeper understanding of our models, to identify biases or errors in our KGs, and most importantly to decide whether our models can be trusted or not.

In this paper we discuss the *Kelpie* framework for explaining embedding-based link predictions, that we originally present in [3]. Accordingly to the taxonomy for explainable AI methods in [4], *Kelpie* is a local post-hoc interpretability method; it can be applied to any LP model based on embeddings, independently of its architecture. Given a prediction, *Kelpie* explains it by identifying the subset of training facts that have enabled it. Given any prediction to explain, *Kelpie* can interpret it in two scenarios, dubbed *necessary* and *sufficient*. A necessary explanation is the minimal set of facts in absence of which the model becomes unable to yield the prediction: in doing so, necessary explanations can suggest why a specific entity has been predicted in a certain way. A sufficient explanation is the minimal set of facts that, if added to other entities, lead the model to yield the same prediction for those entities too: sufficient explanations can thus suggest the rules that would extend the same prediction to any entity. As observed in [5], necessity and sufficiency are the building blocks of any successful explanation. The *Kelpie* code and resources are publicly available.¹

2. Problem definition

A Knowledge Graph (KG) is a labeled directed graph $KG = (\mathcal{E}, \mathcal{R}, \mathcal{G})$ where \mathcal{E} is the set of nodes, or *entities*, in the graph; \mathcal{R} is its set of labels, or *relations*; and $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is its set of edges linking entities via relations, i.e., its set of *facts*. In any fact $\langle h, r, t \rangle$ the first entity h is the *head*, r is the *relation*, and the second entity t is the *tail*.

Models for embedding-based Link Prediction (LP) typically define a *scoring function* ϕ that estimates the plausibility of any fact using the embeddings of its head, relation and tail. In training, models learn embeddings to optimize the plausibility of the known facts: in prediction, new links are identified by finding which entities, if added to incomplete triples as heads or tails, yield the best scores. A *tail prediction* $\langle h, r, t \rangle$ is the process that finds t to be the best scoring tail for the incomplete triple $\langle h, r, ? \rangle$, i.e., $t = \operatorname{argmax}_{e \in \mathcal{E}} \phi(h, r, e)$. Head predictions are defined symmetrically. In the following we focus on tail predictions; all our formulations

¹<https://github.com/AndRossi/Kelpie>

can be adapted for head predictions analogously.

LP literature usually relies on datasets sampled from real-world KGs, and in which the set of extracted facts \mathcal{G} is split into a training set \mathcal{G}_{train} , a validation set \mathcal{G}_{valid} , and a test set \mathcal{G}_{test} . In evaluation, head and tail predictions are performed on each fact in \mathcal{G}_{test} ; in each prediction, the *target entity*, i.e, the expected answer, is ranked against all the others in \mathcal{E} . The obtained ranks are then aggregated into standard global metrics: *Hits@K* (H@K), that measures the fraction of ranks T with value $\leq k$, and *Mean Reciprocal Rank* (MRR), that averages the inverse of all the obtained ranks T . Both H@K and MRR are always between 0 and 1; higher values convey better results. Discussions on the pros and cons of such methodologies can be found in [6, 7, 8].

For some predictions there can exist more than one correct answer (e.g., in case of 1-to- n relations). In this event we follow the common practice called *filtered setting* of excluding such alternative answers when computing the rank of the target entity.

For a given prediction Kelpie defines two explanation types: *necessary* and *sufficient*. We indicate with X the generic *candidate explanation* (either necessary or sufficient) and with X^* the explanation we want to identify and return. We use the subscripts n (e.g., X_n) and s to denote necessary and sufficient explanations respectively. We give definitions of X_n^* and X_s^* w.r.t. tail predictions $\langle h, r, t \rangle$, as head predictions can be handled analogously.

- X_n^* is the smallest set of training facts featuring h such that, removing X_n^* from \mathcal{G}_{train} and retraining the model from scratch, the top ranking tail for $\langle h, r, ? \rangle$ changes to any $e \neq t$.
- X_s^* is the smallest set of training facts featuring h such that, given a set of random entities $C \subseteq \mathcal{E}$ such that t is not the top ranking tail for any $\langle c, r, ? \rangle$ prediction, if we add the facts in X_s^* to any $c \in C$ and retrain the model from scratch, the top ranking tail for $\langle c, r, ? \rangle$ changes to t . When this happens, we say that the c entities have been *converted*.

For instance, in the previous example of the tail prediction $\langle Barack_Obama, nationality, USA \rangle$, the set $\{\langle Barack_Obama, born_in, Honolulu \rangle, \langle Barack_Obama, president, USA \rangle\}$ is a necessary explanation if removing these facts from $Barack_Obama$ the predicted *nationality* changes from USA to any other entity. Analogously, the single-sized set $\{\langle Barack_Obama, president_of, USA \rangle\}$ constitutes a sufficient explanation if adding it to any non-American entities, e.g., $\acute{E}dith_Piaf$, changes their predicted *nationality* to USA .

3. System Overview

The high-level architecture of Kelpie is shown in Figure 2. We briefly describe the Kelpie modules w.r.t. tail predictions $\langle h, r, t \rangle$; head predictions can be handled analogously.

Pre-filter. This module analyzes the set of training facts mentioning h , that we call \mathcal{G}_{train}^h , in order identify and discard the least promising ones: its overall goal is to reduce the search space for the following steps, preventing combinatorial explosion when h is featured in too many facts. The Pre-Filter achieves this goal by computing for each fact in \mathcal{G}_{train}^h a *promisingness* value based on the graph topology, and by returning the subset \mathcal{F}_{train}^h of the top promising facts. Intuitively, topologically closer entities often bear a stronger semantic relationship: so, for any $\langle h, s, q \rangle$ (or $\langle q, s, h \rangle$) connecting h to an entity q close to t we compute *promisingness* as the length of the shortest non-oriented path connecting q to t ; lower values convey better promisingness

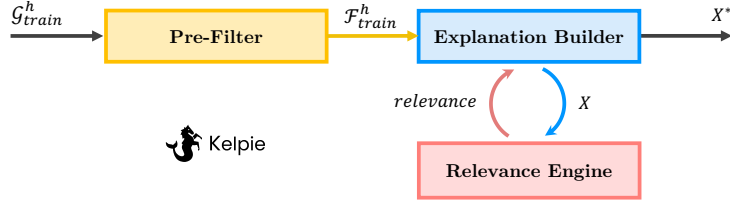


Figure 2: High-level Kelpie architecture, highlighting the interactions among its three modules.

(and thus higher priority in the filtering). Consider again the example in Figure 1 (left) and suppose we want to explain the tail prediction $\langle Barack_Obama, nationality, USA \rangle$:

- $\langle Barack_Obama, president_of, USA \rangle$ has promisingness 0 (the best possible value).
- $\langle Barack_Obama, born_in, Honolulu \rangle$ has promisingness 1.
- $\langle Bill_Gates, supported, Barack_Obama \rangle$ has promisingness 2, as the shortest path from $Bill_Gates$ to USA has length 2: $[\langle Bill_Gates, born_in, Seattle \rangle, \langle Seattle, located_in, USA \rangle]$.

Explanation Builder. This module explores the space of the candidate explanations X that can be obtained by combining the Pre-Filtered facts, with the goal of identifying the smallest X that is relevant enough for the prediction to explain. The notion of *relevance* of an explanation embodies the likelihood of yielding changes in predictions, and it is quantified by the Relevance Engine described in the next paragraph. Suppose that in the example in Figure 1 (left) only the facts $f_1 = \langle Barack_Obama, president_of, USA \rangle$ and $f_2 = \langle Barack_Obama, born_in, Honolulu \rangle$ survive the Pre-Filtering. The Explanation Builder starts by exploring short explanations, such as, $X_1 = \{f_1\}$ and $X_2 = \{f_2\}$; then, it iteratively builds longer ones, such as $X_3 = \{f_1, f_2\}$, until either X^* is found with relevance higher than a user-specified threshold, or early stopping policies are enacted. In the latter case, the best explanation seen so far is returned. Among same-sized explanations (e.g., X_1 and X_2), a prioritization mechanism based on an off-line heuristic is used. For details about early stopping and exploration priority, we refer the reader to Algorithm 3 in [3].

Relevance Engine. This module aims at estimating how adding or removing training facts from h would affect the prediction to explain. Ideally, the relevance of any candidate explanation X could be computed by retraining the model after adding or removing its facts from G_{train} ; in practice this is unfeasible, so Kelpie introduces the novel ML process of *Post-Training* (PT). PT consists in using an already trained model to obtain an alternate version, i.e., a *mimic*, of an entity e and of the corresponding embedding. Given e , PT amounts to: (i) replicating the training facts mentioning e , G_{train}^e , and potentially injecting additions/removals; (ii) training a new embedding to optimize the plausibility of those facts, while keeping all the other embeddings and parameters in the model frozen. Compared to a full training, PT is a lightweight process as it optimizes only one embedding and uses a training set that is several orders of magnitude smaller than the entire G_{train} . For any entity e Kelpie can post-train two types of mimics:

- A *homologous mimic* e' has its embedding initialized randomly, and then post-trained on an exact replica of G_{train}^e . As a result, we expect e' to behave similarly to the original entity e .
- A *non-homologous mimic* e'_{-M} (or e'_{+M}) initialized randomly as well, and then post-trained

		Necessary Explanations Effectiveness										Sufficient Explanations Effectiveness									
		FB15k		WN18		FB15k-237		WN18RR		YAGO3-10		FB15k		WN18		FB15k-237		WN18RR		YAGO3-10	
		$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR	$\Delta H@1$	ΔMRR
TransE	Kelpie	-0.490	-0.321	-0.920	-0.857	-0.540	-0.356	-0.920	-0.904	-0.740	-0.580	-0.319	+0.516	+0.259	-0.434	+0.128	+0.218	-0.117	+0.169	+0.048	+0.109
	DP	-0.380	-0.258	-0.900	-0.859	-0.460	-0.303	-0.770	-0.701	-0.670	-0.533	+0.273	+0.461	+0.183	+0.350	+0.051	+0.080	+0.082	+0.116	+0.036	-0.117
Comp1Ex	Kelpie	-0.850	-0.695	-0.910	-0.827	-0.590	-0.413	-0.980	-0.913	-0.960	-0.858	+0.945	+0.920	+0.953	-0.962	+0.377	+0.490	+0.834	+0.877	+0.858	+0.892
	DP	-0.540	-0.458	-0.800	-0.742	-0.340	-0.185	-0.750	-0.650	-0.810	-0.714	+0.910	+0.888	+0.931	+0.940	+0.245	+0.334	+0.836	+0.878	+0.835	+0.878
	Criagne	-0.030	-0.020	-0.050	-0.045	-0.090	-0.050	-0.180	-0.150	-0.05	-0.030	+0.068	+0.069	+0.105	+0.137	+0.035	+0.038	+0.110	+0.147	+0.000	+0.000
ConvE	Kelpie	-0.710	-0.516	-0.930	-0.860	-0.430	-0.284	-0.980	-0.914	-0.980	-0.884	-0.677	-0.649	-0.903	-0.900	+0.225	-0.203	+0.827	+0.856	+0.799	+0.848
	DP	-0.350	-0.232	-0.790	-0.752	-0.290	-0.195	-0.850	-0.750	-0.880	-0.799	+0.234	+0.199	+0.396	+0.412	+0.202	+0.169	+0.373	+0.419	+0.366	+0.391
	Criagne	-0.080	-0.056	-0.160	-0.146	-0.290	-0.196	-0.170	-0.156	-0.070	-0.042	+0.106	+0.065	+0.164	+0.166	+0.132	+0.094	+0.150	+0.164	+0.019	+0.020

Table 1
Effectiveness of necessary and sufficient explanations

on a set of facts that replicates \mathcal{G}_{train}^e with the removal (or addition) of a set M of facts. After the post-training is over, the mimic is expected to approximate the behaviour that the original entity e would have displayed if the injected variation had been present since the beginning. Examples of homologous and non-homologous mimics are in Figure 1 (center and right respectively). When explaining any tail prediction $\langle h, r, t \rangle$, we can thus estimate the effect of adding (or removing) from h the facts of any candidate explanation X by comparing the outcomes obtained using a homologous mimic h' and a non-homologous mimic h'_{+X} (or h'_{-X}).

4. Experiments

In this section we briefly report the main experimental results of Kelpie. For a deeper report on the experimental evaluation of the system, see [3].

Setup. Our experiments have been run on a server with 88 CPUs Intel Core(TM) i7-3820, 516GB RAM and 4 16GB NVIDIA Tesla P100 GPUs. We consider the 5 best-established dataset in LP literature: FB15k and FB15k-237, sampled from Freebase; WN18 and WN18RR, sampled from WordNet; and YAGO3-10, sampled from YAGO3. We consider 3 models representative for the 3 different families of the taxonomy in [2]: the *geometric* model TransE [9], the *matrix factorization* model Comp1Ex [10], and the *deep learning* model ConvE [11].

End-to-end performance. We compute the performance of Kelpie as the H@1 and MRR variation caused by applying the extracted explanation to the predictions to disable (in the necessary scenario) or to enable (in the sufficient scenario). More specifically:

- In our necessary scenario, for each model and dataset we randomly sample a set of 100 correct test tail predictions $\langle h, r, t \rangle$. We extract their necessary explanations and remove the explanation facts from \mathcal{G}_{train} ; we then re-train the model and check how the removals have worsened the ranks of the target tails t . We measure such worsenings in terms of $\Delta H@1$ and ΔMRR : more negative values correspond to higher effectiveness.
- In our sufficient scenario, we once again sample 100 correct test tail predictions $\langle h, r, t \rangle$ to explain for each model and dataset. We extract their sufficient explanations, each with the

	Necessary Explanations										Sufficient Explanations									
	FB15k		WN18		FB15k-237		WN18RR		YAGO3-10		FB15k		WN18		FB15k-237		WN18RR		YAGO3-10	
	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
TransE	2.78	1.13	2.17	1.24	2.5	1.15	1.67	0.96	1.99	1.20	1.89	1.03	1.02	0.14	3.43	0.83	1.67	0.79	1.45	0.73
CompLEx	3.40	1.10	3.39	1.00	3.83	0.57	3.16	1.08	2.27	1.31	1.40	0.75	1.00	0.00	2.51	1.20	1.00	0.00	1.04	0.31
ConvE	3.36	0.89	2.91	1.20	2.26	1.24	2.66	1.21	1.73	0.97	1.83	1.23	1.01	0.10	3.09	1.10	1.04	0.24	1.18	0.48

Table 2
Lengths of the necessary and sufficient explanations

goal of converting a different set C of 10 random entities. Then, for each prediction $\langle h, r, t \rangle$ we add the explanation facts to all c entities in the corresponding C set, we re-train the model, and we measure how the addition of the explanation facts has improved the tail rank of t in each $\langle c, r, t \rangle$ prediction. We measure such improvements in terms of $\Delta H@1$ and ΔMRR : more positive values correspond to higher effectiveness.

We use as baselines two *data poisoning* systems, namely *Criage* [12] and the framework by Zhang *et al.* [13], that we denote as DP. The goal of these systems is to disable the selected $\langle h, r, t \rangle$ predictions by removing/adding individual training facts. In the necessary scenario we compare directly to their removal setting. In the sufficient scenario, that they do not support natively, we adapt their formulations to identify which facts would not only disable $\langle h, r, t \rangle$ but also enable the selected $\langle c, r, t \rangle$ predictions. We do not run *Criage* on TransE, as the code provided by the authors only supports multiplicative models (e.g., ConvE and CompLEx). Results by *Criage*, DP and Kelpie are reported in Table 1. Kelpie almost always outperforms baselines, by tackling predictions that others fail to explain.

Since *Criage* and DP are limited to explanations with only 1 fact, we have also experimented with a single-fact version of Kelpie; we have obtained results comparable to the best baselines but almost always worse than "full" Kelpie. This demonstrates the effectiveness of post-training at identifying the most relevant facts, as well as the utility of supporting fact combinations.

Explanations Lengths. We report in Table 2, for each scenario, model and dataset, the average (AVG) and the standard deviation (STD) of the lengths of our explanations. We observe that necessary explanations tend to always be longer than sufficient ones, for the same datasets and models. Necessary explanations should encompass *all* the facts supporting a prediction, so that removing them disables the prediction. Sufficient explanations, on the other hand, just need to extend the same prediction to other entities: so, it is usually enough for them to include just a few facts (or even just one) as pieces of evidence for the prediction. We also observe that, in many cases, the average explanation lengths are lesser than 2, suggesting that Kelpie can identify minimal explanations, as expected (extensive experiments on minimality are in [3]).

Explanations and Bias. LP datasets have been found to suffer from various forms of *data bias* [14]; we now report a brief example showing how Kelpie explanations can unveil previously unknown instances of bias in the training data. We have observed that the correctly predicted YAGO3-10 test facts that convey that a person *born_in* a city are generally explained by that person being a soccer player *playing_for* a football team from the same city. This is surprising: in the real world, being member of a football team does not imply having been born in its city.

Guided by our explanations, we have found that, in YAGO3-10, people playing for a football team are born in same city unnaturally often (thus providing data bias) and that personal data are significantly scarce, making birthplace prediction very challenging: the correct predictions, in this regard, seem to be affected even by the slight preference that football players may have towards teams from their birthplace. This type of observations can allow researchers to intervene on LP datasets to make them better adhere to the semantics of the real world.

5. Related Works

So far few works have addressed directly the interpretability of embedding-based LP models. The works most related to ours are Criage [12] and the framework by Zhang *et al.* [13]. Both of them follow a *data poisoning* approach: given a prediction $\langle h, r, t \rangle$, their goal is to find which is the individual fact that, if added to or removed from \mathcal{G}_{train} , worsens the score $\phi(h, r, t)$ the most. Criage [12] uses Influence Functions [15] to estimate how the addition or removal of any fact would affect $\phi(h, r, t)$; unfortunately, the adaptability of their formulation to the scoring functions of non-multiplicative models is unclear. The framework by Zhang *et al.* [13] uses gradient analysis to perturb the embedding of h in the direction that worsens $\phi(h, r, t)$ the most. Other than having empirical differences with Kelpie (as illustrated in Section 4) these methods have an inherently different goal, as they do not aim directly at explaining predictions, but rather they investigate how models withstand adversarial modifications.

One of the aspects that make the interpretability of LP models so challenging is the large variety of ML architectures. In time, researchers have tried to bypass this problem in a variety of ways. Some authors have chosen to support specific architectures [16], while other have proposed inherently interpretable LP models [17]. Finally, a few frameworks explain predictions by focusing on the dataset topology more than on the model and its embeddings [18]. We refer the reader to [3] for further discussion about the application of general purpose explanation methods like LIME [19], SHAP [20] and ANCHOR [21] to the LP setting, and their shortcomings.

6. Conclusions

We have discussed Kelpie, a full-fledged explainability framework for embedding-based LP models. Kelpie explanations highlight the most relevant training samples that have enabled our predictions; they are based on the fundamental notions of necessity and sufficiency, and they can encompass combinations of multiple training samples. In the sparkling topic of LP on Knowledge Graphs, interpretability is a strikingly desirable property; yet, it is hardly ever guaranteed by current state-of-the-art systems and frameworks. The effectiveness of Kelpie explanations, measured through extensive experiments, shows that Kelpie largely surpasses pre-existing methods across almost all scenarios in literature; furthermore, as demonstrated, Kelpie explanations can be precious in identifying unbalances and biases in LP datasets.

Acknowledgments

The work by Donatella Firmani has been supported in part by SEED PNR FLOWER grant 2021.

References

- [1] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, D. Lin, Knowledge base completion via search-based question answering, in: WWW, 2014.
- [2] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, P. Merialdo, Knowledge graph embedding for link prediction: A comparative analysis, TKDD (2021).
- [3] A. Rossi, D. Firmani, P. Merialdo, T. Teofili, Explaining link prediction systems based on knowledge graph embeddings, in: SIGMOD, 2022.
- [4] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, D. Pedreschi, A survey of methods for explaining black box models, ACM Comput. Surv. (2018).
- [5] D. S. Watson, L. Gultchin, A. Taly, L. Floridi, Local explanations via necessity and sufficiency: Unifying theory and practice, in: UAI, 2021.
- [6] A. Rossi, A. Matinata, Knowledge Graph Embeddings: Are Relation-Learning Models Learning Relations?, in: PIE, 2020.
- [7] Y. Wang, D. Ruffinelli, R. Gemulla, S. Broscheit, C. Meilicke, On Evaluating Embedding Models for Knowledge Base Completion, in: RepL4NLP@ACL, 2019.
- [8] A. Rossi, Interpreting link prediction on knowledge graphs., in: SEBD, 2020.
- [9] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: NIPS, 2013.
- [10] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex Embeddings for Simple Link Prediction, in: ICML, 2016.
- [11] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: AAAI, 2018.
- [12] P. Pezeshkpour, Y. Tian, S. Singh, Investigating robustness and interpretability of link prediction via adversarial modifications, in: NAACL-HLT, 2019.
- [13] H. Zhang, T. Zheng, J. Gao, C. Miao, L. Su, Y. Li, K. Ren, Data poisoning attack against knowledge graph embedding, in: IJCAI, 2019.
- [14] A. Rossi, D. Firmani, P. Merialdo, Knowledge graph embeddings or bias graph embeddings? a study of bias in link prediction models, in: DL4KG, 2021.
- [15] P. W. Koh, P. Liang, Understanding black-box predictions via influence functions, in: D. Precup, Y. W. Teh (Eds.), ICML, 2017.
- [16] Z. Ying, D. Bourgeois, J. You, M. Zitnik, J. Leskovec, Gnnexplainer: Generating explanations for graph neural networks, in: NIPS, 2019.
- [17] W. Zhang, S. Deng, H. Wang, Q. Chen, W. Zhang, H. Chen, Xtranse: Explainable knowledge graph embedding for link prediction with lifestyles in e-commerce, in: JIST, 2019.
- [18] W. Zhang, B. Paudel, W. Zhang, A. Bernstein, H. Chen, Interaction embeddings for prediction and explanation in knowledge graphs, in: WSDM, 2019.
- [19] M. T. Ribeiro, S. Singh, C. Guestrin, "Why Should I Trust You?": Explaining the Predictions of Any Classifier, in: SIGKDD, 2016.
- [20] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: NIPS, 2017.
- [21] M. T. Ribeiro, S. Singh, C. Guestrin, Anchors: High-precision model-agnostic explanations, in: AAAI, 2018.