Explaining Completions Produced by Embeddings of Knowledge Graphs

Andrey Ruschel(✉), Arthur Colombini Gusmão, Gustavo Padilha Polleti, and Fabio Gagliardi Cozman

Universidade de São Paulo, São Paulo, SP, Brazil
{andrey.ruschel,fgcozman}@usp.br

Abstract. Advanced question answering typically employs large-scale knowledge bases such as DBpedia or Freebase, and are often based on mappings from entities to real-valued vectors. These mappings, called embeddings, are accurate but very hard to explain to a human subject. Although interpretability has become a central concern in machine learning, the literature so far has focused on non-relational classifiers (such as deep neural networks); embeddings, however, require a whole range of different approaches. In this paper, we describe a combination of symbolic and quantitative processes that explain, using sequences of predicates, completions generated by embeddings.

Keywords: Knowledge graph · Knowledge base · Explainable AI · Embedding · Interpretability

1 Introduction

Query answering systems and chatbots have benefited from symbolic facts stored in knowledge graphs (KGs) such as NELL [16], YAGO [22], Freebase [2]. Even though KGs contain many facts, typically stored as triples "subject, relationship, object", KGs are far from complete, and a broad range of *completion* techniques have emerged recently. These techniques often resort to *embeddings* that turn the symbolic data into quantitative vectors, modeling relations between entities by numeric operations over vectors [19]. Completion of a KG then relies on deciding whether a particular triple is predicted through these numeric operations [25].

While embeddings usually offer the most accurate way to predict relationships between entities, they are rather hard to be interpreted by human users. Consider an example that provides background on what it means to "interpret an embedding". Take, for instance, a chatbot answering the question "Is Paris

The work has been supported by Itaú Unibanco S.A. through the Itaú Scholarship Program (first and third authors are recipients). The last author is partially supported by CNPq grant 312180/2018-7. The work has also been supported by FAPESP grant 2016/18841-0, and in part by the Coordenação de Aperfeiçoamento de Nivel Superior (CAPES) - finance code 001.

G. Kern-Isberner and Z. Ognjanović (Eds.): ECSQARU 2019, LNAI 11726, pp. 324–335, 2019.
https://doi.org/10.1007/978-3-030-29765-7_27

the capital of France?". Suppose the chatbot uses a KG containing countries and cities and the relation *is capital of*, but the triple "Paris, is capital of, France" is not in the KG. And suppose chatbot returns YES: why is it? An acceptable reason might be that another triple shows that Paris is a capital, and Paris as located in France. Another possible explanation to the YES-answer would be one focusing on properties of the embedding itself: we might learn that every time we have a city and a country that map into vectors aligned in some particular direction, the former is the capital of the latter—and that this is happening with Paris and France. Note that the purpose of the latter explanation is to understand the behavior of the embedding when answering a particular question. Both explanations require insights that are more sophisticated than existing techniques to explain classifiers based on detecting which features are most relevant [13,20], as indicating that a particular dimension of an embedding strongly affects the decision does not, in itself, provides any clue as to what the embedding is doing.

In previous work [10], we proposed a framework that can produce explanations for KG completion tasks performed by any embedding model (model-agnostic). In essence, it works by mining significant paths in the KG to build a feature matrix from which explanations are extracted through logistic regression. We present here novel insights and refinements that significantly increase the fidelity of explanations. We start by providing in the next section some background knowledge about KBs and KGs, as well as a brief description of KG completion, graph features and latent features. We then summarize a few notions about interpretability and explanations and present our approach. Later we describe experiments and results and then close the paper with a discussion of possible future work.

2 Background

In this section, we provide a short review of the required concepts about knowledge graphs and knowledge graph completion.

2.1 Knowledge Graphs and Their Completion Tasks

Several large knowledge bases have been created to store information in triples of the form ⟨entity, predicate, entity⟩ (loosely following the RDF framework [23]). One may use also *head* for the subject, *relation* for the predicate, and *tail* for the object. For instance, information about the religion of the king Francis II of the Two Sicilies would be represented as a triple ⟨Francis II of the Two Sicilies, Religion, Catholic⟩. A set of triples can naturally be depicted as a *directed acyclic graph*, with an edge from the head to the tail of a triple.

Large KBs, generally built by extracting facts from unstructured text, suffer from incorrect/incomplete information. A fundamental task that involves KGs is *link prediction*. This is the task of, given a specific entity and a relation, finding a matching entity. For instance, for a given head and relation, to predict the tail $\langle e_h, r, ? \rangle$, or, given a tail and a relation, to predict the matching head

$\langle ?, r, e_t \rangle$. One may be interested instead in finding a relation between two entities $\langle e_h, ?, e_t \rangle$, a challenge sometimes called *relation prediction*. Another task is, given a triple $\langle e_h, r, e_t \rangle$ not previously seen on the KG, to evaluate whether this triple is true or false. This is referred to as *triple classification*. Finally, in *entity resolution* one must detect the same entity in different bits of information. For instance, one can find Barak Obama represented as *Barak Obama* and *B. Obama*.

Completion problems have been investigated within statistical relational learning [7,19]. Some useful notation from that literature will be employed in this paper. Let $\mathcal{E} = \{e_1, ..., e_N\}$ represent the set of all possible entities and $\mathcal{R} = \{r_1, ..., r_M\}$ represent the set of all possible relations in a KG. A possible triple is represented by $x_{h,r,t} = \langle e_h, r, e_t \rangle$, where e_h, r and e_t stand for *head*, *relation* and *tail*, respectively. Note that we use the *open world assumption* in this paper, meaning that facts not present in the KG are only considered unknown. We denote the set of all possible triples (or facts) in $\mathcal{G}$ by $\mathcal{T} = \mathcal{E} \times \mathcal{R} \times \mathcal{E}$.

We now contrast two approaches to KG completion: graph feature models and embedding models.

2.2 Graph Feature Models

Graph feature models aim to perform KG completion by observing characteristics of the graph to infer new facts, often by resorting to rules or similar symbolic manipulation [19].

With the PRA algorithm [11], Lao and Cohen suggested that triples can be predicted by a feature matrix constructed with random walks of bounded length in a KG. Based on PRA, Gardner et al. [6] proposed the *Subgraph Feature Extraction (SFE)* algorithm that we now summarize.

The idea is to focus on a relation r at a time. Denote by $\mathcal{D}^+$ the triples that have r as relation, to emphasize that these are "positive" triples. Note that $\mathcal{D}^+$ depends on the particular r, but we simplify notation by not explicitly referring to r. To train the model, a set of "negative" triples $\mathcal{D}^-$ is built by corrupting positive triples in the KG by randomly replacing one of its entities (head or tail) [25]. For a generic triple $\langle e_h, r, e_t \rangle$ we take a path (a sequence of edges) π from e_h to e_t with at most L edges; each edge in a path corresponds to a relation or the inverse of a relation. A triple is then associated with the set of all path patterns connecting its head to tail. In fact, not all possible paths from head to tail are generated; the SFE algorithm runs random walks to sample such paths.

Denote by $\pi_L(h, r, t)$ a path type of maximum length L connecting entity e_h to e_t. The set of all encountered paths $\pi_L(h, r, t)$ between those entities is represented by $\Pi_L(h, r, t)$. Denote by z_π a binary variable that indicates existence or not of a given path π. The feature vector extracted for a given triple is represented by $\phi_{hrt}^{SFE} = [z_\pi : \pi \in \Pi_L(h, r, t)]$. For a given relation r, using the latter expression, the SFE algorithm constructs a feature matrix combining the feature vectors ϕ_{hrt}^{SFE} extracted for each training example $\langle e_h, r, e_t \rangle \in \{\mathcal{D}^+ \cup \mathcal{D}^-\}$. This feature matrix can be used as input to any classifier; if one chooses a logistic regressor, a parameter matrix w_r is then obtained for the relation r.

We then calculate the probability of existence of the triple $\langle e_a, r, e_b \rangle \notin \mathcal{D}^+$ with $f_{abc}^{SFE} := w_r^T \phi_{abc}^{SFE}$.

To extract paths, the SFE algorithm builds subgraphs departing from each entity e_h and e_t with k steps. If two subgraphs $\mathcal{G}_h$ and $\mathcal{G}_t$ contain paths $\pi_{h,i}$ and $\pi_{t,i}$ departing from each entity and arriving at some intermediate node i, then a path type $\pi_{h,i} \cup \pi_{i,t}$ is stored in the feature vector. Gardner et al. [6] adopted random walks for the SFE algorithm but also proposed to construct the subgraphs via a *breadth-first search (BFS)*, to increase the number of extracted features. To keep the search computationally tractable during BFS in large KGs, they proposed to skip the expansion of nodes with a high out-degree (the number of incoming/outcoming edges). So if a path departing from e_h reaches a node with degree higher than a given number (a parameter of the model), that node will not be expanded in further steps, but it will still be considered as an intermediate node i that can later be merged to the subgraph departing from e_t. This strategy significantly increases the number of extracted features.

2.3 Embedding Models

Latent feature models map semantically rich entities and relations into real-valued vectors. The mappings are referred to as *embeddings*. The state-of-the-art in KG tasks uses this idea because operations and gradients can be easily run in numerical spaces. Usually, an embedding model represents entities as vectors of an arbitrary dimension and relations as operations within the same vector space. The symbolic data is then manipulated through numeric operations over those vectors. The "plausibility" of a triple $x_{h,r,t}$ is represented by a scoring function $f(x_{h,r,t} \mid \Theta)$ [25], where Θ represents the set of parameters of the model. Basically, there are two major families of embeddings: the first one, called *translational distance models*, focus in distance-base scoring functions like *TransE* and its extensions [17,25]. The second is formed of *semantic matching models*, that rely on scoring functions that measure semantic similarity, being *ANALOGY* and *RESCAL* examples of this family [17,25].

Although the interpretability techniques we propose in this paper are agnostic to any particular embedding, we will focus on TransE, a very popular embedding model proposed by Bordes et al. [3], inspired by *Word2Vec* [14]. In TransE, entities and relations are represented by vectors of an arbitrary dimension, and relations are translations within the vector space. A triple $\langle e_h, r, e_t \rangle$ is deemed true when $\boldsymbol{e_h} + \boldsymbol{r} = \boldsymbol{e_t}$ (or rather, when this equality is approximately true within some threshold). The various vectors are obtained by optimization, taking into account all the information in the KG of interest. In short, a vector representation of entities and relations is learned so as to minimize the loss function $||\boldsymbol{e_h} + \boldsymbol{r} - \boldsymbol{e_t}||$ for all facts in the knowledge base.

3 Explaining Embeddings

Currently, embeddings offer the most accurate way to complete KGs, but they are difficult to understand as they strip the underlying KG of its semantic con-

tent. On the other hand, graph feature models capture some of the structure of the KG, thus offering decisions that can be related to semantic properties but that are less accurate than the ones produced by embeddings. Thus one naturally asks whether it is possible to automatically explain completions produced by embeddings, perhaps using the symbolic features of the KG as a source of semantic guidance. This is our strategy in this paper.

First, a word on "Explainable AI", a topic that has received significant attention. Even though there has been work on explaining neural networks since at least the nineties [1,4], the recent emergence of very complex classifiers, for instance, ones based on very deep neural networks or very wide random forests, has led to many insights concerning the interpretation of automated decisions by classifiers [8]. It should be noted that the notion of "interpretability" is not a simple one [12]; it is certainly not an absolute notion as it depends on the target customer. A classifier can be interpreted through mathematical equations if the customer is a data scientist, but it should be explained in a textual manner if the customer is a lawyer in the auditing department. In any practical scenario, one may have "degrees" of interpretability depending on the understanding of causes for the prediction at hand [15]. Also, the interpretation of a model is related to the trust assigned by the user to the model; it is hard to trust a decision that cannot be adequately explained. Another perspective is this: when explaining a prediction, do we want to explain it "absolutely" in the sense that we want to justify why it makes sense, or do we want to explain is "relatively" to the model, focusing on the reasons why the model made the decision even if they are not logically perfect? The former seems useful in all circumstances, but the latter can be even more critical when the intended user is a data scientist trying to figure out the behavior of a classifier, or an auditing specialist trying to determine whether a classifier is biased or not. Simple metrics like accuracy are of no help when interpretability is needed [5]; in fact, there is a natural tension between accuracy and interpretability: more accuracy in the presence of large datasets tends to require more complex models that lead to less interpretable decisions.

There are two distinct basic approaches to interpreting classifiers. First, *decompositional* approaches extract rules and explanations by taking into account the specific structure of the classifier of interest [1]. Second, *pedagogical* or *agnostic* approaches consider the classifier as a black-box and implement a simpler classifier to mimic the outputs from the complex one and also provide explanations about the outputs. In this paper, we focus on agnostic techniques as we wish to provide tools that can be useful for a variety of embedding frameworks.

Interpretations are often generated by detecting which features are most important, through various sensitivity analyses, or perhaps by detecting which data points are most influential [24].

Alas, such approaches cannot work in interpreting embeddings. Indeed, embeddings turn a semantically rich input into numeric vectors, and one cannot operate in the vector space that is actually used in classification. The transfor-

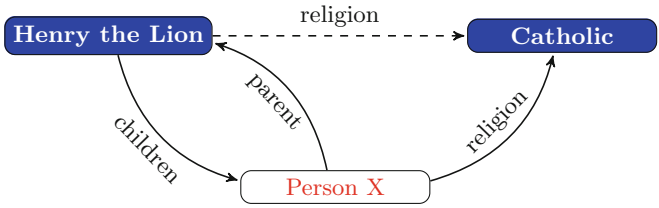

Fig. 1. Example produced from FB13. Entities of interest are in blue; dashed edge is assigned by TransE (other edges belong to FB13). (Color figure online)

mation from semantic entities to vectors is indeed a crucial part of the whole model, and should also be explained.

Our idea is to consider the embedding model we are trying to explain as a black-box and to implement an interpretable classifier around it, by extracting features (path patterns) from the original graph, using the labels predicted by the embedding. This interpretable classifier is then used to produced symbolic explanations, in the form of weighted *Horn clauses*, that are regarded as easily interpretable [19]. The result is a set of symbolic explanations obtained by graph features for each completion produced by the embedding. We pursued this basic idea in a previous publication [10], but our previous proposals had rather low fidelity. Here *fidelity* refers to the fraction of completions where the extracted graph feature model agrees with the original embedding. Of course one should aim at 100% fidelity when interpreting embeddings: there is no point in hiring an "interpreter" that may provide reasons for decisions that were *not* made.

The contribution of this paper is to present a framework able to produce explanations that correctly mimic the embedding classifier for every prediction. Before we plunge into a description of our contributions, it is worth considering a pair of examples generated by our implementation and depicted in Figs. 1 and 2. These examples were generated with data from two popular KGs, namely FB13 [2,21] and NELL186 [9,16]. Each explanation consists of a subgraph containing entities in the KGs; in some cases the specific entity is irrelevant ("Person X", and so on). However, a symbolic explanation can be easily produced: for instance, we see that Henry the Lion is considered catholic by TransE in the FB13 dataset for a simple reason: his child is catholic (a fact that is in FB13). Similarly, Fig. 2 describes why TransE determines UIC Flames to play in the Ice Hockey league using data in NELL186.

We now describe our method in detail; to do so, in this paragraph, we review the XKE-TRUE algorithm by Gusmão et al. [10]. Consider a KG $\mathcal{G}$ with $\mathcal{T} = \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ representing the set of all possible triples of this KG. Denote by $g : \mathcal{T} \to \{0, 1\}$ the function of the embedding black-box classifier. Define $\Pi_{\mathcal{G}}$ as the set of all possible paths connecting two entities, and $P(\Pi_{\mathcal{G}})$ its power set. The feature extraction function performed by the SFE algorithm for a given triple $x_{h,r,t} \in \mathcal{T}$, and for the given graph $\mathcal{G}$, is represented by $SFE : \mathcal{T} \to P(\Pi_{\mathcal{G}})$. The result of applying the SFE algorithm to a triple $x_{h,r,t}$ is denoted by $\Pi_{h,r,t|\mathcal{G}} \in P(\Pi_{\mathcal{G}})$.

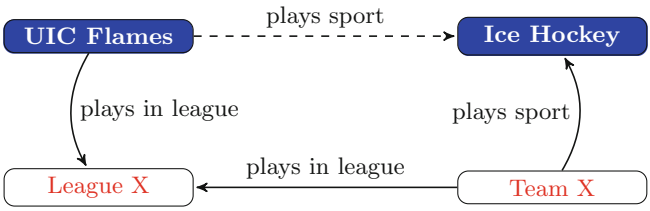

Fig. 2. Example extracted from NELL186. Entities of interest are in blue; the dashed edge is assigned by TransE (other edges belong to NELL186). The explanation for the fact that the team UIC Flames plays Ice Hockey is the fact that they play in a league, and another team that also plays in the same league also plays Ice Hockey. (Color figure online)

Then XKE-TRUE builds an auxiliary training set

$$\mathbb{D} = \{(SFE(x_{h,r,t} \mid \mathcal{G}), g(x_{h,r,t})) \mid x_{h,r,t} \in \mathcal{D}\} \tag{1}$$

and trains a interpretable classifier, in our case a logistic regressor, $g'' : P(\Pi_{\mathcal{G}}) \rightarrow \{0, 1\}$ using $\mathbb{D}$, from which explanations are drawn in the form of weighted Horn clauses, where each rule (feature) is a path type extracted from $\mathcal{G}$, preceded by a weight assigned by the logistic regressor.

Even though the XKE-TRUE algorithm just outlined works reasonably, it has a severe drawback: its fidelity is far from 100%, thus making it fail to provide explanations in many cases. We now present two enhancements to the original XKE-TRUE to improve its fidelity. The first one allows SFE to extract more features from the KG, leading to a substantial increase in fidelity, while the second deals with the case where the interpretable classifier contradicts an embedding prediction.

3.1 Modified SFE

We have implemented a new version of the SFE algorithm, following the same principles proposed by Gardner et al. [6]. Gardner's SFE implementation with *BFS* makes use of a parameter of the model that specifies the maximum node out-degree, and nodes with out-degree above that value will not be expanded. In our BFS implementation, starting nodes with out-degree higher than the maximum value will be expanded only one step away. This single step turned out to be very useful to build many more paths sequences to be used as features.

3.2 XKE-e

Despite the modified SFE algorithm described above, for a reasonable amount of training examples, BFS was not able to find any feature in the KG simply because there was no path of length L connecting the triples entities. This hinders the

fidelity of the resulting logistic regressor. One way to overcome this issue would be to increase L and find paths of greater length, but in large KGs, this would be computationally very expensive. Instead, we propose to leverage the knowledge obtained by applying SFE and logistic regression: for the triples with inconsistent prediction between the logistic regression and the embedding, we build new paths (using the embedding for KG completion) connecting the entities using the most critical paths according to the weights assigned by the logistic regression.

Let us denote by $\mathbb{D}_0$ the subset of $\mathbb{D}$ in which the SFE function applied to the triple resulted in an empty set $\Pi_{h,r,t|\mathcal{G}} = \varnothing$. For each triple $x_{h,r,t} \in \mathbb{D}_0$ we get the active rules w_c of the trained logistic regression and construct this path using $g(\cdot)$, following the algorithm below:

Algorithm 1.1 XKE-e

procedure BUILD-EXTENDED-SET(g, $\mathcal{G}$, $x_{h,r,t}$)
  $\hat{\Pi} \leftarrow \{\}$ ▷ Set of path types for the triple
  $\hat{g} \leftarrow \{\}$ ▷ Set of new found facts
  for all $\pi_{h,r,t} \in \Pi^{(w\neq 0)}_{h,r,t|\mathcal{G}}$ **do**
    for each edge $\in \pi_{h,r,t}$ **do**
      if $g(\pi_{i,j,k}) = 1$ **then** ▷ If edge holds
        $\hat{g} \leftarrow \pi_{i,j,k} \cup \hat{g}$
        if $\pi_{h,r,t} = TRUE$ **then** ▷ If path holds
          $\hat{\Pi} \leftarrow \pi_{h,r,t} \bigcup \hat{\Pi}$
  return $\hat{\Pi}$, $\hat{g}$
procedure XKE-E
  $\Pi_{h,r,t|\mathcal{G}} \leftarrow SFE(x_{h,r,t} \mid \mathcal{G})$
  for all $x_{h,r,t} \in \mathbb{D}_0$ **do**
    $\hat{\Pi}, \hat{g} \leftarrow$ BUILD-EXTENDED-SET(g, $\mathcal{G}, x_{h,r,t}$)
    $\Pi_{h,r,t|\mathcal{G}} \leftarrow \hat{\Pi}$
    $\mathcal{G} \leftarrow \mathcal{G} \cup \hat{g}$
  Using $\mathbb{D}$, train an interpretable classifier $g' : P(\Pi_{\mathcal{G}}) \leftarrow \{0, 1\}$
  Draw explanations from g' in the form of Horn clauses

4 Experiments

Here we present the results obtained with our novel approach, comparing them with results obtained by Gusmão et al. [10]. We will evaluate them according to the following metrics:

- **Accuracy:** ratio of correct predictions by the logistic regressor (logit);
- **Fidelity:** ratio of prediction matches between logit and embedding;
- **F1:** the classical definition, obtained for accuracy and fidelity;
- **Average # of features per example:** is the average number of features extracted by SFE per example in the test set;
- **% of Examples with # of features > 0:** represents the proportion of cases in the test set with at least one feature extracted by SFE;

Table 1. Results (micro-average) for XKE-TRUE/TransE. Results marked with * were extracted from Gusmão et al. [10]. Results highlighted in bold are the best for each dataset.

Dataset	FB13			NELL186	
Embedding accuracy (%)	82.55			86.40	
Maximum path length (SFE)	4*	4	6	4*	4
Expand initial node (SFE)	No	Yes	Yes	No	Yes
Maximum node out-degree (SFE)	100	100	100	100	unl
Avg # of features per example	2.91	4.15	55.89	70.66	78.39
% Examples with # features > 0	54.73	87.09	91.41	50.01	55.47
Explanation mean rules	2.29	3.60	18.86	105.30	102.06
Explanation mean body rule length	3.09	3.01	4.62	3.86	3.86
Fidelity (%)	73.26	75.54	**83.04**	86.55	**88.59**
Accuracy (%)	73.43	75.49	**79.98**	89.10	**92.33**
F1 Fidelity (%)	76.66	76.24	**80.47**	83.19	**86.53**
F1 Accuracy (%)	77.35	77.25	**77.69**	86.89	**91.40**

For interpretability, we consider the following metrics:

- **Explanation Mean # of Rules:** the average number of rules per example that the logistic regressor assigned a weight different than zero;
- **Explanation Mean Body Rule Length:** the average number of relations forming active rules with weight different than zero assigned by the logistic regressor.

Below we describe all model parameters used in our experiments. To fairly compare each SFE strategy, we used the same embeddings trained by Gusmão et al. [10] as proposed by [18]; negative examples were generated via Bernoulli distribution at a 1–1 rate. Model training was limited to 1,000 epochs, splitted into 100 mini-batches and using SGD with Adagrad optimizer. The best model accuracy was obtained with a learning rate $\eta = 1$, ℓ_2 norm, margin $\gamma = 1$ and embedding dimension $k = 100$ for *FB13* and $k = 50$ for *NELL*. For the feature extraction with our implementation of the SFE algorithm, we now describe the parameters. The logistic regressor for each relation was trained using *SGD* to minimize the log loss with elastic net regularization and a grid search was run to find the best fidelity using the following parameters: $\eta = 1$ regularization ratio $\gamma = \{0.1, 0.7, 0.7, 0.9, 0.95, 0.99, 1.0\}$, regularization weight $\alpha = \{0.1, 0.001, 0.0001\}$ and stopping criteria $\epsilon = 0.001$; class weights were inversely proportional to their frequency to properly balance classes.

We can see from Table 1 that relaxing the parameters of the feature extraction deployed by the SFE algorithm helped to increase the number of features; more importantly, this increase led to an improvement in the fidelity results for both datasets. We were able to generate a scenery using paths of length 6 with *FB13*,

which turned out to provide the highest amount of examples with at least one feature, and also the best fidelity in mimicking the embeddings predictions.

For the NELL186 dataset, we found the same positive effect, mainly because we were able to run tests with *unlimited* out-degree. We can see here that the *accuracy* of the interpretable model was better than the accuracy of the embedding model itself, indicating that one could use SFE+logistic regression as a primary tool for KG completion, without the use of the embedding.

As for the interpretability metrics, we saw no significant changes, except for the FB13 scenery with paths of maximum size 6, where the number of active rules per example exploded. In contrast, the explanation mean body rule length only grew from 3 to 4.63, which still can be regarded as easily interpretable.

Table 2. Example of explanation generated by XKE-TRUE, extracted from. [10].

Head	Henry the Lion
Relation	Religion
Tail	Catholic
Reason #1	(0.649) parent^{-1}, religion
Reason #2	(0.500) children, religion
Bias	(0.681)
XKE	0.862
Embedding	1

Table 2 brings the same example from Fig. 1, now showing the active rules obtained by the logistic regressor with its weights, explaining the prediction of the embedding. The intuition behind this explanation is that, by applying the SFE algorithm followed by a logistic regression, XKE-e generalized this fact from the KG attributing a weight for that reason. Indeed it is a much more convincing explanation than a statement saying that de dimension *43* of the entity vectors in $\mathbb{R}^{100}$ was the one that contributed to the correct answer.

5 Conclusion and Future Work

We have presented a novel method to produce explanations for completions generated by embeddings. We started with the XKE-TRUE algorithm [10]; our purpose was to increase the fidelity of that algorithm. We did this by introducing changes to the SFE algorithm and by adding various steps to XKE-TRUE (Algorithm 1.1). The resulting XKE-e algorithm is a novel scheme that has high fidelity, and that produces intuitive and plausible explanations, as we have shown through experiments and through examples.

Despite the advances described here, a significant concern when dealing with operations in knowledge graphs is the exponential growth of possible paths with

the number of entities. In future work, we would like to investigate local explanations through the extraction of more expressive paths, as we believe that this would even further improve the fidelity of our explanations.

References

1. Andrews, R., Diederich, J., Tickle, A.B.: Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowl.-Based Syst. **8**, 373–389 (1995)
2. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: ACM SIGMOD International Conference on Management of Data, pp. 1247–1250 (2008)
3. Bordes, A., Usunier, N., García-Durán, A., Weston, J.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, pp. 2787–2795 (2013)
4. Craven, M.W., Shavlik, J.W.: Extracting thee-structured representations of thained networks. In: Advances in Neural Information Processing Systems, pp. 24–30 (1996)
5. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. Harvard J. Law Technol. **31**, 841–887 (2017)
6. Gardner, M., Mitchell, T.: Efficient and expressive knowledge base completion using subgraph feature extraction. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1488–1498. Association for Computational Linguistics (2015)
7. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning). The MIT Press, Cambridge (2007)
8. Gunning, D.: Broad Agency Announcement Explainable Artificial Intelligence (XAI). Technical report (2016)
9. Guo, S., Wang, Q., Wang, B., Wang, L., Guo, L.: Semantically Smooth Knowledge Graph Embedding. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, pp. 84–94 (2015)
10. Gusmão, A.C., Correia, A.C., De Bona, G., Cozman, F.G.: Interpreting embedding models of knowledge bases : a pedagogical approach. In: 2018 ICML Workshop on Human Interpretability in Machine Learning (WHI 2018), pp. 79–86, no. Whi (2018)
11. Lao, N., Cohen, W.W.: Relational retrieval using a combination of path-constrained random walks. Mach. Learn. **81**(n.1), 53–67 (2010)
12. Lipton, Z.C.: The mythos of model interpretability. In: ICML Workshop on Human Interpretability in Machine Learning, pp. 96–100 (2016)
13. Lundberg, S.M., Allen, P.G., Lee, S.I.: A unified approach to interpreting model predictions. In: Advances in Neural Information Processing Systems, pp. 4765–4774 (2017)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3, 1–12 (2013)
15. Miller, T.: Explanation in artificial intelligence: insights from the social sciences. Artif. Intell. **267** (2017)
16. Mitchell, T., et al.: Never-ending learning. Commun. ACM **61**(1), 2302–2310 (2015)

17. Nguyen, D.Q.: An overview of embedding models of entities and relationships for knowledge base completion. CoRR abs/1703.0 (2017)
18. Nguyen, D.Q., Sirts, K., Qu, L., Johnson, M.: Neighborhood mixture model for knowledge base completion. In: Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, Berlin, Germany, pp. 40–50 (2016)
19. Nickel, M., Murphy, K., Tresp, V., Gabrilovich, E.: A review of relational machine learning for knowledge graphs. IEEE **104**, 11–33 (2015)
20. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should i trust you?": explaining the predictions of any classifier. In: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)
21. Socher, R., Chen, D., Manning, C.D., Ng, A.Y.: Reasoning with neural tensor networks for knowledge base completion. In: Neural Information Processing Systems (2003), pp. 926–934 (2013)
22. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a core of semantic knowledge unifying WordNet and Wikipedia. In: WWW 2007 (2007)
23. W3: RDF 1.1 Concepts and Abstract Syntax. https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/
24. Wachter, S., Mittelstadt, B., Russell, C.: Couterfactual explanations without opening the black-box: automated decisions and the GDPR. Harvard J. Law Technol. **31**, 841 (2017)
25. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding : a survey of approaches and applications. IEEE Trans. Knowl. Data Eng. **29**(12), 2724–2743 (2017)