[Technical Report] ExCut: Explainable Embedding-based Clustering over Knowledge Graphs

Mohamed H. Gad-Elrab[1,2], Daria Stepanova[2], Trung-Kien Tran[2], Heike Adel[2], and Gerhard Weikum[1]

[1] Max-Planck Institute for Informatics, Saarland Informatics Campus, Germany
{gadelrab,weikum}@mpi-inf.mpg.de
[2] Bosch Center for Artificial Intelligence, Renningen, Germany
{firstname.lastname}@de.bosch.com

Abstract. Clustering entities over knowledge graphs (KGs) is an asset for explorative search and knowledge discovery. KG embeddings have been intensively investigated, mostly for KG completion, and have potential also for entity clustering. However, embeddings are latent and do not convey user-interpretable labels for clusters. This work presents ExCut, a novel approach that combines KG embeddings with rule mining methods, to compute informative clusters of entities along with comprehensible explanations. The explanations are in the form of concise combinations of entity relations. ExCut jointly enhances the quality of entity clusters and their explanations, in an iterative manner that interleaves the learning of embeddings and rules. Experiments on real-world KGs demonstrate the effectiveness of ExCut for discovering high-quality clusters and their explanations.

1 Introduction

Motivation. Knowledge graphs (KGs) are collections of triples of the form $\langle subject\ predicate\ object \rangle$ used for important tasks such as entity search, question answering and text analytics, by providing rich repositories of typed entities and associated properties. For example, Tedros Adhanom is known as a health expert, director of the World Health Organization (WHO), alumni of the University of London, and many more.

KGs can support analysts in exploring sets of interrelated entities and discovering interesting structures. This can be facilitated by **entity clustering**, using unsupervised methods for grouping entities into informative subsets. Consider, for example, an analyst or journalist who works on a large corpus of topically relevant documents, say on the Coronavirus crisis. Assume that key entities in this collection have been spotted and linked to the KG already. Then the KG can guide the user in understanding what kinds of entities are most relevant. With thousands of input entities, from health experts, geolocations, political decision-makers all the way to diseases, drugs, and vaccinations, the user is likely overwhelmed and would appreciate a group-wise organization. This task of computing entity clusters [4,6,16] is the problem we address.

Merely clustering the entity set is insufficient, though. The user also needs to understand the nature of each cluster. In other words, clusters must be explainable, in the form of **user-comprehensible labels**. As entities have types in the KG, an obvious solution is to label each cluster with its prevalent entity type. However, some KGs

have only coarse-grained types and labels like "people" or "diseases" cannot distinguish health experts from politicians or virus diseases from bacterial infections. Switching to fine-grained types, such as Wikipedia categories, on the other hand, causes the opposite problem: each entity is associated with tens or hundreds of types, and it is unclear which of these would be a good cluster label. The same holds for an approach where common SPO properties (e.g., $educatedIn$ `UK`) are considered as labels. Moreover, once we switch from a single KG to a set of linked open data (LOD) sources as a joint entity repository, the situation becomes even more difficult.

Problem Statement. Given a large set of entities, each with a substantial set of KG properties in the form of categorical values or relations to other entities, our problem is to jointly tackle: (i) **Clustering:** group the entities into k clusters of semantically similar entities; (ii) **Explanation:** generate a user-comprehensible concise labels for the clusters, based on the entity relations to other entities.

State-of-the-Art and its Limitations. The problem of clustering relational data is traditionally known as conceptual clustering (see, *e.g.*, [27] for overview). Recently, it has been adapted to KGs in the Semantic Web community [6,16]. Existing approaches aim at clustering graph-structured data itself by, *e.g.*, introducing novel notions of distance and similarity directly on the KG [4,5]. Due to the complexity of the data, finding such universally good similarity notions is challenging [5]. Moreover, existing relational learning approaches are not sufficiently scalable to handle large KGs with millions of facts, *e.g.*, YAGO [28] and Wikidata [31]. Clustering entities represented in latent space, *e.g.*, [12,32], helps to overcome this challenge, yet, the resulting clusters are lacking explanations, clustering process is prone to the embedding quality, and hyperparameters are hard to tune [5]. Explaining clusters over KGs, *e.g.*, [29] focuses on the discovery of explanations for given perfect clusters. However, obtaining those in practice is not straightforward.

Approach. To address the above shortcomings, we present **ExCut**, a new method for computing explainable clusters of large sets of entities. The method uses KG embedding as a signal for finding plausible entity clusters, and combines it with logical rule mining, over the available set of properties, to learn interpretable labels. The labels take the form of concise conjunctions of relations that characterize the majority of entities in a cluster. For example, for the above Coronavirus scenario, we aim at mining such labels as $worksFor(X, Y) \wedge type(Y, \texttt{health\_org}) \wedge hasDegreeIn(X, \texttt{life\_sciences})$ for a cluster of health experts, $type(X, \texttt{disease}) \wedge causedBy(X, Y) \wedge type(Y, \texttt{virus})$ for a cluster of virus diseases, and more. A key point in our approach is that these labels can in turn inform the entity embeddings, as they add salient information. Therefore, we interleave and iterate the computation of embeddings and rule mining adapting the embeddings using as feedback the information inferred by the learned rules.

Contributions. Our main contributions can be summarized as follows:

- We introduce ExCut, a novel approach for computing explainable clusters, which combines embedding-based clustering with symbolic rule learning to produce human-understandable explanations for the resulting clusters. These explanations can also serve as new types for entities.

– We propose several strategies to iteratively fine-tune the embedding model to maximize the explainability and accuracy of the discovered clusters based on the feedback from the learned explanations.
– We evaluate ExCut on real-world KGs. In many cases, it out-performs state-of-the-art methods w.r.t. both clustering and explanations quality.

ExCut's code, data and technical report are available at github.com/mhmgad/ExCut.

2 Preliminaries

Knowledge Graphs. KGs represent interlinked collections of factual information, encoded as a set of $\langle subject\ predicate\ object \rangle$ triples, *e.g.*, $\langle$`tedros_adhanom` *directorOf* `WHO`$\rangle$. For simplicity, we write triples as in predicate logics format, e.g., *directorOf*(`tedros_adhanom`, `WHO`). A signature of a KG $\mathcal{G}$ is $\Sigma_{\mathcal{G}} = \langle \mathbf{P}, \mathbf{E} \rangle$, where $\mathbf{P}$ is a set of binary predicates and $\mathbf{E}$ is a set of entities, *i.e.*, constants, in $\mathcal{G}$.

KG Embeddings. KG embeddings aim at representing all entities and relations in a continuous vector space, usually as vectors or matrices called *embeddings*. Embeddings can be used to estimate the likelihood of a triple to be true via a scoring function: $f : \mathbf{E} \times \mathbf{P} \times \mathbf{E} \rightarrow \mathbb{R}$. Concrete scoring functions are defined based on various vector space assumptions:

(i) The translation-based assumption, *e.g.*, TransE [1] embeds entities and relations as vectors and assumes $\mathbf{v_s} + \mathbf{v_r} \approx \mathbf{v_o}$ for true triples, where $\mathbf{v_s}, \mathbf{v_r}, \mathbf{v_o}$ are vector embeddings for subject s, relation r and object o, resp.
(ii) The linear map assumption, *e.g.*, ComplEx [30] embeds entities as vectors and relations as matrices. It assumes that for true triples, the linear mapping $\mathbf{M_r}$ of the subject embedding $\mathbf{v_s}$ is close to the object embedding $\mathbf{v_o}$: $\mathbf{v_s}\mathbf{M_r} \approx \mathbf{v_o}$.

The likelihood that these assumptions of the embedding methods hold should be higher for triples in the KG than for those outside. The learning process is done through minimizing the error induced from the assumptions given by their respective loss functions.

Rules. Let $\mathbf{X}$ be a set of variables. A *rule* r is an expression of the form $head \leftarrow body$, where $head$, or $head(r)$, is an atom over $\mathbf{P} \cup \mathbf{E} \cup \mathbf{X}$ and $body$, or $body(r)$, is a conjunction of positive atoms over $\mathbf{P} \cup \mathbf{E} \cup \mathbf{X}$. In this work, we are concerned with *Horn rules*, a subset of first-order logic rules with only positive atoms, in which every head variable appears at least once in the body atoms.

Example 1. An example of a rule over the KG in Fig. 1 is $r : has(X,$`covid19`$) \leftarrow worksWith(X,Y), has(Y,$`covid19`$)$ stating that coworkers of individuals with `covid19` infection, potentially, also have `covid19`.

Execution of rules over KGs is defined in the standard way. More precisely, let $\mathcal{G}$ be a KG, r a rule over $\Sigma_{\mathcal{G}}$, and a an atom, we write $r \models_{\mathcal{G}} a$ if there is a variable assignment that maps all atoms of $body(r)$ into $\mathcal{G}$ and $head(r)$ to the atom a. *Rule-based inference* is the process of applying a given rule r on $\mathcal{G}$, which results in the extension $\mathcal{G}_r$ of $\mathcal{G}$ defined as: $\mathcal{G}_r = \mathcal{G} \cup \{a \mid r \models_{\mathcal{G}} a\}$.

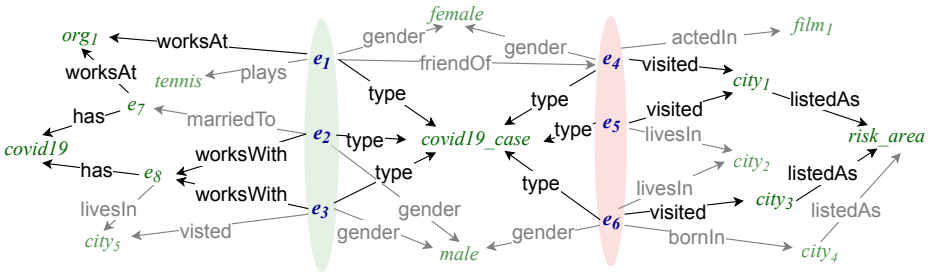


Fig. 1: An example KG with potential COVID-19 cases split into two entity clusters (in green and red). Black edges are relevant for the potential explanations of these clusters.

Example 2. Application of the rule r from Example 1 on the KG $\mathcal{G}$ from Figure 1 results in $r \models_{\mathcal{G}} has(\mathtt{e}_2, \mathtt{covid19})$ and $r \models_{\mathcal{G}} has(\mathtt{e}_3, \mathtt{covid19})$. Hence, $\mathcal{G}_r = \mathcal{G} \cup \{has(\mathtt{e}_2, \mathtt{covid19}), has(\mathtt{e}_3, \mathtt{covid19})\}$.

3 Model for Computing Explainable Clusters

Given a KG, a subset of its entities and an integer k, our goal is to find a "good" split of entities into k clusters and compute explanations for the constructed groups that would serve as informative cluster labels. E.g., consider the KG in Fig. 1, the set of target entities $\{\mathtt{e}_1, \ldots, \mathtt{e}_6\}$ and the integer $k = 2$. One of the possible solutions is to put $\mathtt{e}_{1-3}$ into the first cluster C_1 and the other three entities into the second one C_2. Explanations for this split would be that C_1 includes those who got infected via interacting with their coworkers, while the others were infected after visiting a risk area. Obviously, in general there are many other splits and identifying the criteria for the best ones is challenging. Formally, we define the problem of *computing explainable entity clusters* as follows:

Definition 1 (Computing Explainable Entity Clusters Problem).
Given: *(i) a knowledge graph $\mathcal{G}$ over $\Sigma_{\mathcal{G}} = \langle \mathbf{P}, \mathbf{E} \rangle$; (ii) a set $T \subseteq \mathbf{E}$ of target entities; (iii) a number of desired clusters $k > 1$; (iv) an explanation language L; and (v) an explanation evaluation function $d : 2^L \times 2^T \times \mathcal{G} \rightarrow [0..1]$*
Find: *a split $\mathcal{C} = \{C_1, \ldots C_k\}$ of entities in T into k clusters and a set of explanations $\mathcal{R} = \{r_1, \ldots, r_k\}$ for them, where $r_i \in L$, s.t. $d(\mathcal{R}, \mathcal{C}, \mathcal{G})$ is maximal.*

3.1 Explanation Language

Explanations (i.e., informative labels) for clusters can be characterized as conjunctions of common entity properties in a given cluster; for that Horn rules are sufficient. Thus, our explanation language relies on (cluster) explanation rules defined as follows:

Definition 2 (Cluster Explanation Rules). *Let $\mathcal{G}$ be a KG with the signature $\Sigma_{\mathcal{G}} = \langle \mathbf{P}, \mathbf{E} \rangle$, let $C \subseteq \mathbf{E}$ be a subset of entities in $\mathcal{G}$,* i.e.*, a cluster, and $\mathbf{X}$ a set of variables. A* (cluster) explanation rule *r for C over $\mathcal{G}$ is of the form*

$$r: \; belongsTo(X, \mathtt{e}_C) \leftarrow p_1(\vec{X_1}), \ldots, p_m(\vec{X_m}), \tag{1}$$

where $\mathtt{e}_C \notin \mathbf{E}$ *is a fresh unique entity representing the cluster* C*,* $belongsTo \notin \mathbf{P}$ *is a fresh predicate, and* $body(r)$ *is a finite set of atoms over* $\mathbf{P}$ *and* $\mathbf{X} \cup \mathbf{E}$*.*

Example 3. A possible explanation rule for $C_1 = \{\mathtt{e}_1, \mathtt{e}_2, \mathtt{e}_3\}$ in $\mathcal{G}$ from Fig. 1 is

$$r : belongsTo(X, \mathtt{e}_{\mathtt{C}_1}) \leftarrow worksWith(X, Y), has(Y, \mathtt{covid19})$$

which describes C_1 as a set of people working with infected colleagues.

Out of all possible cluster explanation rules we naturally prefer **succinct** ones. Therefore, we put further restrictions on the explanation language L by limiting the number of rule body atoms (an adjustable parameter in our method).

3.2 Evaluation Function

The function d from Def. 1 compares solutions to the problem of explainable entity clustering w.r.t. their quality, and ideally d should satisfy the following two criteria:

(i) **Coverage:** Given two explanation rules for a cluster, the one covering more entities should be preferred and
(ii) **Exclusiveness:** Explanation rules for different clusters should be (approximately) mutually exclusive.

The coverage measure from data mining is a natural choice for satisfying (i).

Definition 3 (Explanation Rule Coverage). *Let* $\mathcal{G}$ *be a KG,* C *a cluster of entities, and* r *a cluster explanation rule. The* coverage *of* r *on* C *w.r.t.* $\mathcal{G}$ *is*

$$cover(r, C, \mathcal{G}) = \frac{|\{c \in C | r \models_{\mathcal{G}} belongsTo(c, \mathtt{e}_C)\}|}{|C|} \tag{2}$$

Example 4. Consider clusters $C_1 = \{\mathtt{e}_1, \mathtt{e}_2, \mathtt{e}_3\}$, $C_2 = \{\mathtt{e}_4, \mathtt{e}_5, \mathtt{e}_6\}$ shown in Fig. 1. The set of potential cluster explanation rules along with their coverage scores for C_1 and C_2 respectively, is given as follows:

$r_1 : belongsTo(X, \mathtt{e}_{\mathtt{C}_i}) \leftarrow type(X, \mathtt{covid19\_case})$	1	1
$r_2 : belongsTo(X, \mathtt{e}_{\mathtt{C}_i}) \leftarrow gender(X, \mathtt{male})$	0.67	0.33
$r_3 : belongsTo(X, \mathtt{e}_{\mathtt{C}_i}) \leftarrow worksWith(X, Y), has(Y, \mathtt{covid19})$	0.67	0
$r_4 : belongsTo(X, \mathtt{e}_{\mathtt{C}_i}) \leftarrow visited(X, Y), listedAs(Y, \mathtt{risk\_area})$	0	1

While addressing (i), the coverage measure does not account for the criteria (ii). Indeed, high coverage of a rule for a given cluster does not imply a low value of this measure for other clusters. For instance, in Example 4 r_1 is too general, as it perfectly covers entities from both clusters. This motivates us to favour (approximately) mutually exclusive explanation rules, *i.e.*, explanation rules with high coverage for a given cluster but low coverage for others (similar to [13]). To capture this intuition, we define the *exclusive explanation coverage* of a rule for a cluster given other clusters as follows.

Definition 4 (Exclusive Explanation Rule Coverage). *Let $\mathcal{G}$ be a KG, let $\mathcal{C}$ be a set of all clusters of interest, $C \in \mathcal{C}$ a cluster, and r an explanation rule. The* exclusive explanation rule coverage *of r for C w.r.t. $\mathcal{C}$ and $\mathcal{G}$ is defined as*

$$exc(r,C,\mathcal{C},\mathcal{G}) = \begin{cases} 0, \text{ if } \min\limits_{C' \in \mathcal{C} \setminus C} \{cover(r,C,\mathcal{G}) - cover(r,C',\mathcal{G})\} \leq 0 \\ cover(r,C,\mathcal{G}) - \frac{\sum\limits_{C' \in \mathcal{C} \setminus C} cover(r,C',\mathcal{G})}{|\mathcal{C} \setminus C|}, \text{ otherwise.} \end{cases} \quad (3)$$

Example 5. Consider $\mathcal{C} = \{C_1, C_2\}, \mathcal{R} = \{r_1, r_2, r_3, r_4\}$ from Example 4 and the KG $\mathcal{G}$ from Fig. 1. We have $exc(r_1, C_1, \mathcal{C}, \mathcal{G}) = exc(r_1, C_2, \mathcal{C}, \mathcal{G}) = 0$, which disqualifies r_1 as an explanation for either of the clusters. For r_2, we have $exc(r_2, C_1, \mathcal{C}, \mathcal{G}) = 0.34$ making it less suitable for the cluster C_1 than r_3 with $exc(r_3, C_1, \mathcal{C}, \mathcal{G}) = 0.67$. Finally, r_4 perfectly explains C_2, since $exc(r_4, C_2, \mathcal{C}, \mathcal{G}) = 1$.

Similarly, we can measure the *quality* of a collection of clusters with their explanations by averaging their per-cluster exclusive explanation rule coverage.

Definition 5 (Quality of Explainable Clusters). *Let $\mathcal{G}$ be a KG, $\mathcal{C} = \{C_1, \ldots, C_k\}$ a set of entity clusters, and $\mathcal{R} = \{r_1, \ldots, r_k\}$ a set of cluster explanation rules, where each r_i is an explanation for C_i, $1 \leq i \leq k$. The* explainable clustering quality *q of $\mathcal{R}$ for $\mathcal{C}$ w.r.t. $\mathcal{G}$ is defined as follows:*

$$q(\mathcal{R}, \mathcal{C}, \mathcal{G}) = \frac{1}{|\mathcal{C}|} \sum_{i=1}^{|\mathcal{C}|} exc(r_i, C_i, \mathcal{C}, \mathcal{G}) \quad (4)$$

Realizing function d in Definition 1 by the above measure allows us to conveniently compare the solutions of the explainable clusters discovery problem.

Example 6. Consider $\mathcal{G}$ from Fig. 1, the set of target entities $T = \{\mathtt{e}_1, \ldots, \mathtt{e}_6\}$, $k = 2$, language L of cluster explanation rules with at most 2 body atoms, and the evaluation function d given as q from Def. 5. The best solution to the respective explainable entity clustering problem is $\mathcal{C} = \{C_1, C_2\}$, $\mathcal{R} = \{r_3, r_4\}$, where C_1, C_2, r_3, r_4 are from Example 4. We have that $q(\mathcal{R}, \mathcal{C}, \mathcal{G}) = 0.83$.

4 Method

We now present our method ExCut, which iteratively utilizes KG *Embedding-based Clustering* and *Rule Learning* to compute explainable clusters. More specifically, as shown in Fig. 2, ExCut starts with (1) *Embedding Learning* for a given KG. Then, it performs (2) *Clustering* of the entities in the target set over the learned embeddings. Afterwards, (3) *Rule Learning* is utilized to induce explanation rules for the constructed clusters, which are ranked based on the exclusive coverage measure. Using the learned explanation rules, we perform (4) *Rule-based Inference* to deduce new entity-cluster assignment triples reflecting the learned structural similarities among the target entities. Then, ExCut uses the rules and the inferred assignment triples in constructing feedback to guide the clustering in the subsequent iterations. We achieve that by fine-tuning the embeddings of the target entities in Step (5) *Embedding Adaptation*.

In what follows we present the detailed description of ExCut's components.

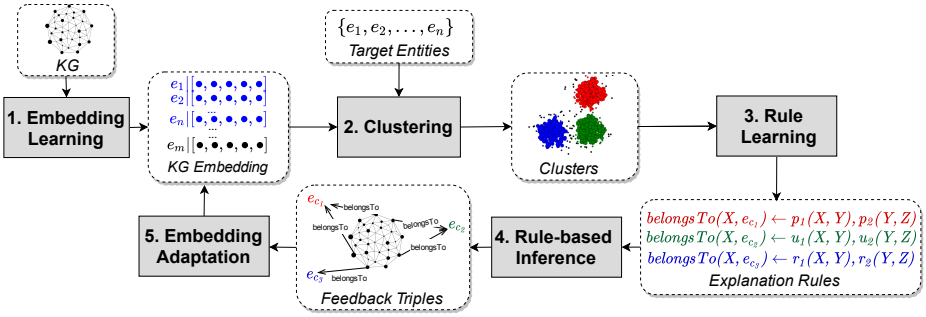


Fig. 2: ExCut pipeline overview.

4.1 Embedding Learning and Clustering

Embedding Learning. ExCut starts with learning vector representations of entities and relations. We adopt KG embeddings in this first step, as they are well-known for their ability to capture semantic similarities among entities, and thus could be suited for defining a robust similarity function for *clustering relational data* [5]. Embeddings are also effective for dealing with data incompleteness, *e.g.*, predicting the potentially missing fact $worksWith(\texttt{e}_1, \texttt{e}_7)$ in Fig. 1. Moreover, embeddings facilitate the inclusion of unstructured external sources during training, *e.g.*, textual entity descriptions [33].

Conceptually, any embedding method can be used in our approach. We experimented with TransE [1] and ComplEx [30] as prominent representatives of translation-based and linear map embeddings. To account for the context surrounding the target entities, we train embeddings using the whole KG.

Clustering. The *Clustering* step takes as input the trained embedding vectors of the target entities and the number k of clusters to be constructed. We perform clustering relying on the embeddings as features to compute pairwise distances among the target entities using standard distance functions, *e.g.*, *cosine distance*. Various classical clustering approaches (see *e.g.*, [18] for overview) or more complex embedding-driven clustering techniques [32] could be exploited in this step. In this paper, we rely on the traditional *Kmeans* method [17] as a proof of concept.

For KGs with types, the majority of embedding models [1,30] would map entities of a certain type to similar vectors [32]. For example, $\texttt{e}_1$ and $\texttt{e}_2$ in Fig. 3.A are likely to be close to each other in the embedding space, and thus have a high chance of being clustered together. An ideal embedding model for explainable clustering should follow the same intuition even if types in the KG are missing. In other words, it should be capable of assigning similar vectors to entities that belong to structurally similar subgraphs of certain pre-specified complexity. For instance, in Fig. 3.B, both $\texttt{e}_1$ and $\texttt{e}_2$ belong to subgraphs reflecting that these entities are married to politicians with some `covid19` symptom, and hence should be mapped to similar vectors.

Despite certain attempts to consider specific graph patterns (*e.g.*, [15]), to the best of our knowledge none of the existing embedding models is general enough to capture patterns of arbitrary complexity. We propose to tackle this limitation (see Sec. 4.3) by passing to the embedding model feedback created using cluster explanation rules learned in the Step 3 of ExCut.

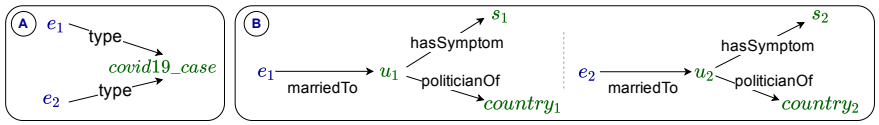


Fig. 3: KG fragments.

4.2 Explanation Mining

KG-based Explanations. KG embeddings and the respective clusters constructed in Steps 1 and 2 of our method are not interpretable. However, since KG embeddings are expected to preserve semantic similarities among entities, the clusters in the embedding space should intuitively have some meaning. Motivated by this, in ExCut, we aim at decoding these similarities by learning rules over the KG extended by the facts that reflect the cluster assignments computed in the *Clustering* step.

Rule Learning Procedure. After augmenting $\mathcal{G}$ with the facts $belongsTo(\mathtt{e}, \mathtt{e}_{\mathtt{C}_i})$ for all entities $\mathtt{e}$ clustered in C_i, we learn Horn rules of the form (1) from Def. 2. The need for our own dedicated rule learning procedure comes from the fact that the state-of-the-art methods such as AMIE+ [8], AnyBurl [19], and RuDiK [23] are designed to learn closed rules, *i.e.*, each variable or constant in rule head appears exactly once in it's body, while we do not introduce such requirement. In addition, the quality measurements and search heuristics of existing systems are optimized towards learning rules for link prediction, which is not our target. Following [8], we model rules as sequences of atoms, where the first atom is the head of the rule (i.e., $belongsTo(X, \mathtt{e}_{\mathtt{C}_i})$ with C_i being the cluster to be explained), and other atoms form the rule's body.

For each cluster C_i, we maintain an independent queue of intermediate rules, initialized with a single rule atom $belongsTo(X, \mathtt{e}_{\mathtt{C}_i})$, and then exploit an iterative breadth-first search strategy. At every iteration, we expand the existing rules in the queue using the following *refinement operators*:

(i) *add a positive dangling atom*: add a binary positive atom with one fresh variable and another variable appearing in the rule, *i.e.*, *shared variable* , *e.g.*, adding $worksAt(X, Y)$, where Y is a fresh variable not appearing in the current rule;
(ii) *add a positive instantiated atom*: add a positive atom with one argument being a constant and the other one a shared variable , *e.g.*, adding $locatedIn(X, usa)$, where usa is a constant, and X appears elsewhere in the rule constructed so far.

These operators produce a set of new rule candidates, which are then filtered relying on the given explanation language L. Suitable rules with a minimum coverage of 0.5, *i.e.*, the rules covers the majority of the respective cluster, are added to the output set. We refine the rules until the maximum length specified in the language bias is reached. Finally, we rank the constructed rules based on the *exclusive explanation coverage* (Def. 4), and select the top m rules for each cluster.

Example 7. Assume that for $\mathcal{G}$ in Fig. 1, and $T = \{\mathtt{e}_1, \ldots, \mathtt{e}_6\}$, the embedding-based clustering resulted in the following clusters $C_1 = \{\mathtt{e}_1, \mathtt{e}_2, \underline{\mathtt{e}_4}\}$ and $C_2 = \{\mathtt{e}_5, \mathtt{e}_6, \underline{\mathtt{e}_3}\}$, where $\mathtt{e}_4$ and $\mathtt{e}_3$ are incorrectly placed in wrong clusters. The top cluster explanation rules for C_2 ranked based on exc measure from Def. 4 are:

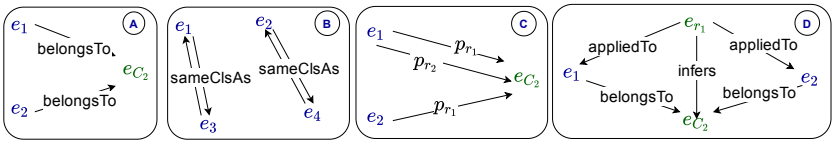


Fig. 4: Inferred clusters assignment triples modeling options.

$r_1 : belongsTo(X, \mathtt{e}_{C_2}) \leftarrow visited(X, Y)$ 0.67
$r_2 : belongsTo(X, \mathtt{e}_{C_2}) \leftarrow gender(X, \mathtt{male})$ 0.33
$r_3 : belongsTo(X, \mathtt{e}_{C_2}) \leftarrow visited(X, Y), listedAs(Y, \mathtt{risk\_area}).$ 0.33

Inferring Entity-Clusters Assignments. In the *Rule-based Inference* (Step 4 in Fig. 2), we apply the top-m rules obtained in the *Rule Learning* step on the KG to predict the assignments between the target entities and the discovered clusters over $belongsTo$ relation using standard deductive reasoning techniques. The computed assignment triples are ranked and filtered based on the *exc* score of the respective rules that inferred them.

Example 8. Application of the rules from Ex. 7 on $\mathcal{G}$ w.r.t. the target entities $\mathtt{e}_{1-6}$ results in the cluster assignment triples: $\{belongsTo(\mathtt{e}_3,\mathtt{e}_{C_2}), belongsTo(\mathtt{e}_4,\mathtt{e}_{C_2}), belongsTo(\mathtt{e}_2, \mathtt{e}_{C_2})\}$. Note that based on r_1, $\mathtt{e}_4$ is assigned to C_2 instead of C_1.

4.3 Embedding Adaptation

Learned explanation rules capture explicit structural similarities among the target entities. We propose to utilize them to create feedback to guide the embedding-based clustering towards better explainable clusters. This feedback is passed to the embedding model in the form of additional training triples reflecting the assignments inferred by the learned rules. Our intuition is that such added triples should potentially help other similarities of analogous nature to be discovered by the embeddings, compensating for the embedding-based clustering limitation discussed in Section 4.1.

Specifically, the embedding adaptation (Step 5 in Fig. 2) is summarized as follows:

(a) From the *Rule Learning* and *Rule-based Inference* steps, described above, we obtain a set of *cluster assignment triples* of the form $belongsTo(e, e_C)$ together with rules inferring them, where e is an entity in the input KG $\mathcal{G}$ and e_C is a new entity uniquely representing the cluster C.
(b) We then model the cluster assignments from (a) and rules that produce them using one of our four strategies described below and store the results in $\mathcal{G}^{inf}$.
(c) A subset $\mathcal{G}^{context}$ of $\mathcal{G}$ consisting of triples that surround the target entities is then constructed.
(d) Finally, we fine-tune the embedding model by training it further on the data compiled from $\mathcal{G}^{inf}$ and $\mathcal{G}^{context}$.

Modeling Rule-based Feedback. Determining the adequate structure and amount of training triples required for fine-tuning the embedding model is challenging. On the one hand, the training data should be rich enough to reflect the learned structure, but

on the other hand, it should not corrupt the current embedding. We now present our proposed four strategies for representing the inferred cluster-assignments along with the corresponding rules as a set of triples $\mathcal{G}^{inf}$ suitable for adapting the embedding. The strategies are listed in the ascending order of their complexity.

- **Direct**: As a straightforward strategy, we directly use the inferred entity-cluster assignment triples in $\mathcal{G}^{inf}$ as shown in Fig. 4.A, *e.g.*, $belongsTo(\mathtt{e}_1, \mathtt{e}_{\mathtt{C}_2})$.
- **Same-cluster-as**: In the second strategy, we model the inferred assignments as edges only. As shown in Fig. 4.B, we compile $\mathcal{G}^{inf}$ using triples of $sameClsAs$ relations between every pair of entities belonging to the same cluster as the learned rules suggest, *e.g.*, $sameClsAs(\mathtt{e}_1, \mathtt{e}_2)$. Modeling the cluster assignments using fresh relations allows us to stress the updates related to the target entities, as no extra entities are added to the KG in this strategy.
- **Rules as edges**: Third, we propose to model the inferred assignments together with the rules which led to their prediction. More precisely, for every rule r which deduced $belongsTo(\mathtt{e}, \mathtt{e}_{\mathtt{C}_i})$, we introduce a fresh predicate p_r and add a triple $p_r(\mathtt{e}, \mathtt{e}_{\mathtt{C}_i})$ to the training set $\mathcal{G}^{inf}$, as illustrated in Fig. 4.C. This allows us to encode all conflicting entity-cluster assignments (*i.e.*, assignments, in which an entity belongs to two different clusters) and supply the embedding model with richer evidence about the rules that predicted these assignments.
- **Rules as entities**: Rules used in the deduction process can also be modeled as entities. In the fourth strategy, we exploit this possibility by introducing additional predicates $infers$ and $appliedTo$, and for every rule r a fresh entity $\mathtt{e}_r$. Here, each $belongsTo(\mathtt{e}, \mathtt{e}_{\mathtt{C}_i})$ fact deduced by the rule r is modeled in $\mathcal{G}^{inf}$ with two triples $infers(\mathtt{e}_\mathtt{r}, \mathtt{e}_{\mathtt{C}_i})$ and $appliedTo(\mathtt{e}_\mathtt{r}, \mathtt{e})$ as shown in Fig. 4.D.

Embedding Fine-Tuning. At every iteration i of ExCut, we start with the embedding vectors obtained in the previous iteration $i-1$ and train the embedding further with a set of adaptation triples $\mathcal{G}^{adapt}$. The set $\mathcal{G}^{adapt}$ is composed of the union of all $\mathcal{G}^{inf}_j$ for $j = 1 \dots i$ and a set of context triples $\mathcal{G}^{context}$. For $\mathcal{G}^{context}$, we only consider those directly involving the target entities as a subject or object. E.g., among the facts in the surrounding context of $\mathtt{e}_1$, we have $worksAt(\mathtt{e}_1, \mathtt{org}_1)$ and $plays(\mathtt{e}_1, \mathtt{tennis})$.

Our empirical studies (see the technical report[3]) showed that including assignment triples from previous iterations $j < i$ leads to better results; thus, we include them in $\mathcal{G}^{adapt}$, but distinguish entity and relation names from different iterations. Additionally, considering the context subgraph helps in regulating the change caused by the cluster assignment triples by preserving some of the characteristics of the original embeddings.

5 Extended Experiments

We evaluate the effectiveness of ExCut in producing explainable clusters. Experimental results show that in many cases, ExCut out-performing the state-of-the-art methods

[3] Code, data and the technical report are available at https://github.com/mhmgad/ExCut.

Table 1: Datasets statistics

	UWCSE	WebKB	Terror.	IMDB	Mutag.	Hepatitis	LUBM	YAGO
# Target Entities	209	106	1293	268	230	500	2850	3900
# Target Clusters	2	4	6	2	2	2	2	3
# KG Entities	991	5906	1392	578	6196	6511	242558	4295825
# Relations	12	7	4	4	14	19	22	38
# Facts	2216	72464	17117	1231	30805	77585	2169451	12430700

regarding both clustering and explanation qualities. Specifically, we report the experiments covering the following aspects: (i) the quality of the clusters produced by ExCut compared to existing clustering approaches; (ii) the quality of the computed explanations for discovered the clusters using external metric; (iii) the benefits of interleaving embedding and rule learning for enhancing the quality of the clusters and their explanations; and (iv) the impact of using different embedding paradigms, feedback modelings and including the context triples and the feedback history from previous iterations.

5.1 ExCut Implementation and Configuration

We implemented ExCut in Python and configured its components as follows:

KG Embedding. We adapted off-the-shelf embedding library Ampligraph [3] to allow model fine-tuning. We experimented with two of the provided embedding models: *TransE* [1] and *ComplEx* [30] Each of them is a representative for translation-based and linear-map models, respectively, and conceptually any embedding model or pre-trained model can be used.

We set the size of the embedding vectors to 100 and train a base model using the whole KG for 100 epochs, using stochastic gradient descent optimizer with a learning rate of 0.0005. During the fine-tuning of embedding model, we further train the model for 25 epochs with a learning rate of 0.005.

Clustering. ExCut uses the implementation offered by *scikit-learn* lib., and as a proof of concept, we demonstrate the results of using *K-means* clustering in all configurations. In addition, in this extended version, we report on using DBSCAN, Spectral, and Hierarchical clustering.

Rule Learning. We implemented the algorithm described in Sec 4.2. For experiments, we fix the language bias of the mined explanations to paths of length two, *e.g.*, $belongsTo(x, C_i) \leftarrow p(x, y), q(y, z)$ where z is either a free variable or bind to a constant. The explanations covers all patterns described in Figure **??**.

Modeling Rule-based Feedback . We experiment with the four modeling strategies introduced in the main paper. We refer to these configurations as follows: direct (*belongToCl*), same cluster as edges (*sameClAs*), rules as edges (*entExplCl*),and rules as entities (*followExp*).

5.2 Experiment Setup

Datasets. We performed experiments on the following datasets, which are widely used for relational clustering [4]: *UWCSE*, *WebKB*, *Terrorists*, *IMDB*, *Mutagenesis*, *Hepatities*. Since those datasets are relatively small, we additionally experimented with the following large-scale datasets:

(i) *LUBM-Courses*: a subset of entities from LUBM syntactic KG [9] which consists of facts simulating the university domain. Target entities are distributed over two clusters: `graduate_course` and `undergraduate_course`; and
(ii) *Yago-Artwork*: a set of target entities randomly selected from *Yago* KG [28]. The entities are uniformly distributed over the following three classes: `book`, `song`, and `movie`. To avoid trivial explanations, type triples for target entities were removed from the KG.

The statistics of the datasets is summarized in Table 1.

Baselines. We compare ExCut to the following clustering methods:

(i) *ReCeNT* [4] is a state-of-the-art relational clustering approach, which clusters entities relying on a similarity score computed based on the structure of the graph;
(ii) *Deep EmbeddingClustering (DEC)* is an embedding-based clustering method that performs dimension reduction jointly with clustering using auto-encoder network and Kullback-Leibler divergence loss function to guide the model; and
(iii) Standard *Kmeans* applied directly over the embedding: *TransE (Kmeans-T)* and *ComplEx (Kmeans-C)*. This base line is is equivalent to using ExCut without iterations.

Clustering Quality Metrics. We measure the clustering quality with respect to the ground truth classes using three standard metrics: *Micro Accuracy (ACC)*, *Adjusted Rand Index (ARI)*, and *Normalized Mutual Information (NMI)*. The higher are the values for the respective metrics, the better is the clustering. *Besides, in this extended version, we report also the Homogeneity score (HMS)*. We also compute the average *Silhouette width (Sil)* [26] which represents the compactness of a single cluster vs the separation between different clusters.

Explanations Quality Metrics. The quality of the generated explanations is measured using the coverage metrics defined in main paper, namely, *per cluster coverage (Cov)* and *exclusive coverage (Exc)*. In addition, we adapted the "novelty" metric *Weighted Relative Accuracy (WRA)* [14], which represents a trade-off between the coverage and the accuracy of the discovered explanations. We compute the average of the respective quality of the top explanations for all clusters, and by comparing this value to the ground truth we evaluate the quality of the solution to the explainable clustering problem found by ExCut.

All experiments were conducted on 80 core Linux machine using 500GB RAM. In all experiments, reported results are the average over 5 runs.

5.3 Experiment Results

In seven out of eight datasets, our approach outperforms the baselines with regard to the overall clustering and explanation quality metrics. Additionally, the quality of the computed explanations increases after few iterations.

Clustering Quality. Table 2 presents the quality of clustering for the state-of-the-art clustering approaches, in the first 2 rows, followed by ExCut first without any feedback as a baseline then with the four feedback strategies, where ExCut-T and ExCut-C stand for ExCut with TransE and ComplEx respectively.

For all datasets except for *Mutagensis*, ExCut achieved, in general, better results w.r.t. the *ARI* value than the state-of-the-art methods. Furthermore, ExCut-T results in significantly better clusters compared to Kmeans-T, *i.e.*, the direct application of *Kmeans* on the TransE embedding model on all datasets apart from *Terrorists*. The reason is that the latter contains mainly attributed predicates, and hence requires a different cluster explanation rule language bias.

Our system managed to fully re-discover the ground truth clusters for the two datasets: *UWCSE* and *IMDB*. The accuracy enhancement by ExCut-T compared to the respective baseline (Kmeans-T) exceeds 30% for *IMDB* and *Hepatitis*. Other quality measurements indicate similar increments.

Explanation Quality. Table 3 shows the average quality of the top explanations for the discovered clusters. More specifically, the average coverage per cluster (*Cov*) and the average exclusive coverage (*Exc*) are intrinsic evaluation metrics used as our optimization functions, while *WRA* is the extrinsic one.

The last row presents the quality of the learned explanations for the ground truth clusters; these values are not necessarily 1.0, as perfect explanations under the specified language bias may not exist. We report them as reference points.

ExCut enhances the average quality of explanations for the discovered clusters compared to the ones obtained by the baselines. Similar to clustering quality, for *UWCSE* and *IMDB* our method achieved the explanations quality of the ground truth. For other datasets, our method obtained higher explanations quality than the respective baselines. This demonstrates the effectiveness of the proposed feedback mechanism in adapting the embedding model to better capture the graph structures in the input KGs.

Results on large-scale KGs. In Table 4 we present quality measures for clustering and explainability of ExCut running with TransE on *LUBM* and *YAGO*. ExCut succeeds to compute the ground truth clusters on *LUBM*. Moreover, despite the noise in *YAGO*, our system achieves an average of 40% enhancement in the clustering accuracy. The quality of the explanations is also improved. ReCent did not scale on *LUBM* and *YAGO* due to memory requirements.

To demonstrate the explainability of ExCut's results, in Table 5, we present the top-3 explanations for each generated cluster along with their quality on the *YAGO* KG. In the ground truth, C_1, C_2, C_3 are clusters for entities of the type `song`, `book`, and `movie` respectively. One can observe that the explanations generated by ExCut-T are more intuitive and of higher quality than those obtained using Kmeans-T. The correlation between the explanation relevance and the used quality metrics can also be observed.

5.4 Results Analysis

In **Enhancements over Iterations**. Fig. 5, we present a sample for the quality of the clusters and the aggregated quality of their top explanations over 10 iterations of ExCut-T using the *followExpl* configuration. In general, clustering and explanations qualities consistently improved over the iterations, which demonstrates the advantage of the introduced embedding fine-tuning procedure. For *IMDB*, the qualities drop at the beginning, but increase and reach highest values at the third iteration. This highlights the benefit of accumulating the auxiliary triples for enhancing the feedback signal, thus preventing the embedding tuning from diverging. The charts also show a correlation between the clustering and explanation quality, which proves our hypothesis that the introduced exclusive explanation rule coverage measure (*Exc*) is useful for selecting a good clustering.

ExCut Configurations. With respect to the effects of different embeddings and feedback modeling, as shown in Tables 2 and 3, we observe that ExCut with *TransE* is more robust than with *ComplEx* regardless of the feedback modeling method. Furthermore, modeling the feedback using *followExpl* strategy leads to better results on the majority of the datasets. This reflects the benefits of passing richer feedback to the embedding, as it allows for better entity positioning in the latent space.

Silhouette Width. In Table 6, we report the *average silhouette width* score for all datasets. By comparing the results of basic *Kmeans-T* and *Kmeans-C* against the results ExCut, we observe that for all datasets except Yago, ExCut enhanced the silhouette width significantly. This emphasises the benefit of the rule-based feedback in achieving more *cluster-friendly* embedding model. In addition, the *silhouette width* enhanced more with ExCut's *sameClAs* configuration, which reflects the effect of the dense feedback. Finally, LUBM and Yago did not follow the same behaviour as other datasets; yet, *silhouette width* is still above Zero. One reason is that these datasets contain many long-tail entities that did not have either enough context triples or clusters-assignments triples

Impact of the context and history on embedding adaptation. Table 7 shows the clustering quality and explanations quality resulting from including the context and/or assignment triples from previous iterations while adapting the embedding models. The results illustrate the effectiveness of including either the context or history or both of them in enhancing the results for all datasets.

5.5 Usefulness of the explanations

To assess the usefulness and human-understandability of the generated explanation rules, we conducted the following user study. The intuition of the user-study is to observe how well the humans can identify the clusters from the explanations.

Task. We provided the participant with:

1. Several clusters (groups) of entities. Each cluster is represented with some sample entities (*e.g.*, three entities) belonging to this cluster.
2. For each sample entity, we provided a short summary about the entity and, if available, a link to the Wikipedia page.

3. A set of different potential explanations (patterns), including, as baselines, frequent patterns among all entities and patterns with high coverage within each specific cluster. Besides, we provide top explanations generated by ExCut. Provided patterns are represented in natural language for ease of readability by the end-user.

We asked the participants to match each pattern of all relevant clusters (*i.e.*, zero or more).

Metric. A *useful* explanation is the one that is *exclusively matched* to the correct cluster by the participants. Therefore, for every *explanation-cluster* pair, we compute the ratio of responses where the pair is *exclusively matched*. Let $match(r_i, c_m) = 1$ if the user matched explanation r_i to the cluster c_m, and $match(r_i, c_m) = 0$ otherwise. Thus, the explanation r_i is *exclusively matched* to the cluster c_m if additionally, $match(r_i, c_j) = 0 \; \forall j \neq m$.

Execution. We published a Mechanical Turk task using the explanation learned on the Yago dataset. We showed the participants the three cluster Movies, Books, and Songs without showing them the names. For every cluster, we showed three prominent examples from the dataset, *e.g.*, "Harry Potter" as a book. We showed them a total of ten explanations, consisting of the top explanations in Table 5 and some baseline explanations. We provided the participants with two examples to ensure a clear understanding of the task and to filter our random responses.

Results. Fig. 6 summarizes the results of the 50 responses collected via the user-study. Each bar shows the ratio of responses exclusively matching explanation r_i to each of the provided clusters. The results show that the majority of the participants exclusively matched explanations r_3 and r_{10} to *movies*; r_7 and r_9 to *books*; and r_6 and r_8 to *songs*. The explanations r_3, r_6, and r_9 have been learned by ExCut. The high relative exclusive matching ratio to the corresponding correct cluster for the ExCut explanations demonstrates their usefulness in differentiating between the given clusters.

Furthermore, in Figure 7, we report the ratio of the explanations-to-cluster matches, *i.e.*, including both exclusive and non-exclusive matches. The results demonstrate that the above-mentioned descriptions are correctly matched to the corresponding clusters by the participants even when in the case of a confusion (*i.e.*, non-exclusive matches).

6 Related Work

Clustering relational data has been actively studied (*e.g.*, [4,6,7,16,27]). The majority of the existing approaches are based on finding interesting features in KGs and defining distance measures between their vectors. Our work is conceptually similar, but we let embedding model identify the features implicitly instead of computing them on the KG directly, which is in spirit of linked data propositionalization [25].

A framework for explaining given high-quality clusters using linked data and inductive logic programming has been proposed in [29]. While [29] aims at explaining existing clusters, we are focused on performing jointly clustering and explanation learning in an iterative fashion to discover clusters of high quality as well as their explanations. The work [12] targets interpreting embedding models by finding concept spaces in node embeddings and linking them to a simple external type hierarchy. This is different from

our method of explaining clusters computed over embedding models by learning rules from a given KG. Bouraoui et al. [2] proposed a method for learning conceptual space representations of known concepts by associating a Gaussian distribution over a learned vector space with each concept. In [10,24] the authors introduce methods for answering logical queries over the embedding space. In contrast, in our work, the concepts/queries are not given, but need to be discovered.

While the step of explanation learning in our method is an adaptation of [8], the extension of other exact symbolic rule learning methods [19,23] is likewise possible. As our main goal is to explain simultaneously several clusters computed over the embedding model relying exclusively on the structural properties of KGs, we did not consider employing neural-based approximate rule learning techniques such as [21,22].

Several methods recently focused on combining [11,34] and comparing [5,20] rule learning and embedding methods. The authors of [11] propose to rank rules learned from KGs by relying both on their embedding-based predictive quality and traditional rule measures, which is conceptually different from our work. In [34] an iterative method for joint learning of linear-map embeddings and OWL axioms (without nominals) has been introduced. The triples inferred by the learned axioms are injected into the KG, before the embedding is re-trained from scratch in the subsequent iteration. In contrast, the rule-based feedback that we generate is not limited to only fact predictions, but encodes further structural similarities across entities. Furthermore, we do not re-train the whole model from scratch, but rather adapt the context triples of target entities accounting for the feedback. Finally, unlike [34], the rules that we learn support constants, which allow to capture a larger variety of explanations.

7 Conclusion

We have proposed ExCut, an approach for explainable KG entity clustering, which iteratively utilizes embeddings and rule learning methods to compute accurate clusters and human-readable explanations for them. Our approach is flexible, as any embedding model can be used. Experiments show the effectiveness of ExCut on real-world KGs.

There are several directions for future work. Considering more general rules (*e.g.*, with negations) in the *Rule Learning* component of our method or exploiting several embedding models instead of a single one in the *Embedding-based Clustering* step should lead to cleaner clusters. Further questions to study include the analysis of how well our method performs when the number of clusters is very large, and how the feedback from the rules can be used to determine the number of clusters automatically.

References

1. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS, pp. 2787–2795 (2013)
2. Bouraoui, Z., Schockaert, S.: Learning conceptual space representations of interrelated concepts. In: IJCAI. pp. 1760–1766 (2018)
3. Costabello, L., Pai, S., Van, C.L., McGrath, R., McCarthy, N., Tabacof, P.: AmpliGraph: a Library for Representation Learning on Knowledge Graphs (Mar 2019)

4. Dumancic, S., Blockeel, H.: An expressive dissimilarity measure for relational clustering over neighbourhood trees. MLJ (2017)
5. Dumancic, S., García-Durán, A., Niepert, M.: On embeddings as an alternative paradigm for relational learning. CoRR **abs/1806.11391** (2018)
6. Fanizzi, N., d'Amato, C., Esposito, F.: Conceptual clustering and its application to concept drift and novelty detection. In: ESWC. pp. 318–332 (2008)
7. Fonseca, N.A., Costa, V.S., Camacho, R.: Conceptual clustering of multi-relational data. In: ILP. pp. 145–159 (2011)
8. Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.M.: Fast rule mining in ontological knowledge bases with amie++. The VLDB Journal **24**(6), 707–730 (2015)
9. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmark for OWL knowledge base systems. J. Web Semant. **3**(2-3), 158–182 (2005)
10. Hamilton, W.L., Bajaj, P., Zitnik, M., Jurafsky, D., Leskovec, J.: Embedding logical queries on knowledge graphs. In: NeurIPS 2018. pp. 2030–2041 (2018)
11. Ho, V.T., Stepanova, D., Gad-Elrab, M.H., Kharlamov, E., Weikum, G.: Rule learning from knowledge graphs guided by embedding models. In: ISWC. pp. 72–90 (2018)
12. Idahl, M., Khosla, M., Anand, A.: Finding interpretable concept spaces in node embeddings using knowledge bases. In: Cellier, P., Driessens, K. (eds.) ML/KDD. pp. 229–240 (2020)
13. Knobbe, A.J., Ho, E.K.Y.: Pattern teams. In: PKDD. pp. 577–584 (2006)
14. Lavrač, N., Flach, P., Zupan, B.: Rule evaluation measures: A unifying view. In: ILP. pp. 174–185 (1999)
15. Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., Liu, S.: Modeling relation paths for representation learning of knowledge bases. In: EMNLP. pp. 705–714 (2015)
16. Lisi, F.A.: A pattern-based approach to conceptual clustering in FOL. In: ICCS. vol. 4068, pp. 346–359 (2006)
17. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Symp. on math. stat. and prob. vol. 1, pp. 281–297 (1967)
18. Madhulatha, T.S.: An overview on clustering methods. CoRR **1205.1117** (2012)
19. Meilicke, C., Chekol, M.W., Ruffinelli, D., Stuckenschmidt, H.: Anytime bottom-up rule learning for knowledge graph completion. In: IJCAI. pp. 3137–3143 (2019)
20. Meilicke, C., Fink, M., Wang, Y., Ruffinelli, D., Gemulla, R., Stuckenschmidt, H.: Fine-grained evaluation of rule- and embedding-based systems for knowledge graph completion. In: ISWC. pp. 3–20 (2018)
21. Omran, P.G., Wang, K., Wang, Z.: Scalable rule learning via learning representation. In: IJCAI. pp. 2149–2155 (2018)
22. Omran, P.G., Wang, Z., Wang, K.: Knowledge graph rule mining via transfer learning. In: PAKDD. pp. 489–500 (2019)
23. Ortona, S., Meduri, V.V., Papotti, P.: Robust discovery of positive and negative rules in knowledge bases. In: ICDE. pp. 1168–1179. IEEE (2018)
24. Ren, H., Hu, W., Leskovec, J.: Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In: ICLR (2020)
25. Ristoski, P., Paulheim, H.: A comparison of propositionalization strategies for creating features from linked open data. In: 1st W. on LD for Knowledge Disc. (2014)
26. Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics **20**, 53 – 65 (1987)
27. Suárez, A.P., Mart'inez Trinidad, J.F., Carrasco-Ochoa, J.A.: A review of conceptual clustering algorithms. Artif. Intell. Rev. **52**(2), 1267–1296 (2019)
28. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A Core of Semantic Knowledge. In: Proceedings of WWW. pp. 697–706 (2007)
29. Tiddi, I., d'Aquin, M., Motta, E.: Data patterns explained with linked data. In: ECML/PKDD. pp. 271–275 (2015)

30. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML. pp. 2071–2080 (2016)
31. Vrandecic, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014)
32. Wang, C., Pan, S., Hu, R., Long, G., Jiang, J., Zhang, C.: Attributed graph clustering: A deep attentional embedding approach. In: IJCAI. pp. 3670–3676 (2019)
33. Xie, R., Liu, Z., Jia, J., Luan, H., Sun, M.: Representation learning of KGs with entity descriptions. In: AAAI. pp. 2659–2665 (2016)
34. Zhang, W., Paudel, B., Wang, L., Chen, J., Zhu, H., Zhang, W., Bernstein, A., Chen, H.: Iteratively learning embeddings and rules for knowledge graph reasoning. In: WWW. pp. 2366–2377 (2019)

Table 2: Clustering results of ExCut compared to the baselines

	Methods	UWCSE				IMDB				Hepatitis				Mutagenesis				WebKB				Terrorist			
		ACC	ARI	HMS	NMI	ACC	ARI	HMS	NMI	ACC	ARI	HMS	NMI	ACC	ARI	HMS	NMI	ACC	ARI	HMS	NMI	ACC	ARI	HMS	NMI
SOA	ReCeNT	0.895	0.597	0.486	0.542	0.605	0.024	0.017	0.013	0.506	-0.007	0.007	0.007	0.774	0.297	0.239	0.236	0.519	0.000	0.000	-0.250	0.365	0.098	0.154	0.127
	DEC	0.671	0.166	0.125	0.122	0.541	0.004	0.012	0.008	0.545	0.007	0.010	0.010	0.514	-0.004	0.002	0.002	0.310	0.025	0.059	0.054	0.369	0.163	0.324	0.255
ExCut-T Kmeans	Kmeans-T	0.909	0.659	0.514	0.514	0.583	0.026	0.108	0.075	0.512	-0.003	0.000	0.000	0.517	-0.004	0.003	0.003	0.333	0.006	0.056	0.055	0.534	0.328	0.554	0.443
	belongToCl	0.989	0.957	0.926	0.919	1.000	1.000	1.000	1.000	0.828	0.427	0.335	0.353	0.675	0.119	0.127	0.126	0.429	0.128	0.185	0.174	0.517	0.268	0.384	0.313
	sameClAs	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.559	0.010	0.006	0.006	0.648	0.083	0.079	0.078	0.360	0.055	0.080	0.079	0.354	0.028	0.068	0.060
	entExplCl	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.820	0.406	0.314	0.331	0.638	0.071	0.083	0.082	0.425	0.134	0.214	0.197	0.451	0.17	0.279	0.227
	followExpl	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.822	0.412	0.318	0.333	0.643	0.078	0.084	0.084	0.440	0.149	0.237	0.217	0.453	0.163	0.272	0.220
ExCut-C Kmeans	Kmeans-C	0.542	-0.004	0.009	0.008	0.525	-0.002	0.005	0.004	0.516	-0.001	0.000	0.000	0.730	0.207	0.180	0.178	0.489	0.205	0.366	0.335	0.509	0.234	0.333	0.278
	belongToCl	0.962	0.849	0.775	0.767	1.000	1.000	1.000	1.000	0.633	0.073	0.049	0.049	0.730	0.209	0.184	0.182	0.505	0.229	0.401	0.369	0.542	0.257	0.341	0.287
	sameClAs	0.978	0.912	0.859	0.858	1.000	1.000	1.000	1.000	0.576	0.021	0.017	0.017	0.732	0.212	0.185	0.182	0.379	0.081	0.186	0.174	0.340	0.029	0.095	0.083
	entExplCl	0.968	0.875	0.829	0.813	0.646	0.082	0.269	0.186	0.693	0.147	0.11	0.109	0.732	0.212	0.187	0.185	0.515	0.237	0.385	0.356	0.526	0.246	0.355	0.294
	followExpl	0.993	0.973	0.946	0.943	1.000	1.000	1.000	1.000	0.659	0.104	0.075	0.076	0.727	0.203	0.178	0.175	0.508	0.224	0.371	0.340	0.517	0.242	0.347	0.289
ExCut-T Spectral	Spectral-T	0.947	0.796	0.726	0.707	0.612	0.049	0.243	0.168	0.502	-0.002	0.000	0.000	0.504	-0.004	0.000	0.000	0.329	0.085	0.110	0.101	0.460	0.286	0.498	0.391
	belongToCl	0.967	0.868	0.822	0.804	1.000	1.000	1.000	1.000	0.656	0.095	0.086	0.085	0.657	0.094	0.096	0.095	0.457	0.185	0.292	0.267	0.441	0.122	0.181	0.146
	entExplCl	0.967	0.868	0.822	0.804	1.000	1.000	1.000	1.000	0.654	0.093	0.089	0.089	0.665	0.105	0.109	0.109	0.506	0.205	0.277	0.255	0.494	0.115	0.158	0.137
	followExpl	0.967	0.868	0.822	0.804	1.000	1.000	1.000	1.000	0.690	0.141	0.093	0.094	0.652	0.089	0.080	0.079	0.300	0.031	0.073	0.066	0.450	0.167	0.261	0.210
ExCut-T DBSCAN	DBSCAN-T	0.703	0.000	0.000	0.000	0.881	0.000	0.000	0.000	0.588	0.000	0.000	0.000	0.587	-0.006	0.001	0.002	0.519	0.000	0.000	0.000	0.415	0.060	0.213	0.190
	belongToCl	0.751	0.279	0.207	0.189	0.903	0.256	0.136	0.210	0.588	0.000	0.000	0.000	0.600	0.000	0.000	0.000	0.519	0.000	0.000	0.000	0.405	-0.011	0.018	0.020
	entExplCl	0.957	0.885	0.898	0.808	0.881	0.000	0.000	0.000	0.588	0.000	0.000	0.000	0.600	0.000	0.000	0.000	0.519	0.000	0.000	0.000	0.351	0.028	0.201	0.154
	followExpl	0.967	0.909	1.000	0.900	0.907	0.294	0.159	0.239	0.588	0.000	0.000	0.000	0.600	0.000	0.000	0.000	0.519	0.000	0.000	0.000	0.390	-0.027	0.030	0.034
ExCut-T Heir.	Heir-T	0.555	-0.043	0.020	0.021	0.675	0.027	0.006	0.004	0.546	0.002	0.026	0.027	0.500	-0.005	0.001	0.001	0.379	-0.025	0.049	0.053	0.442	0.209	0.386	0.311
	belongToCl	0.967	0.868	0.822	0.804	0.627	0.062	0.182	0.126	0.694	0.148	0.100	0.100	0.617	0.051	0.050	0.049	0.449	0.099	0.144	0.147	0.355	0.092	0.178	0.145
	entExplCl	0.967	0.868	0.822	0.804	0.791	0.074	0.013	0.012	0.726	0.202	0.140	0.141	0.574	-0.014	0.010	0.016	0.527	0.215	0.232	0.245	0.306	0.062	0.143	0.117
	followExpl	0.971	0.886	0.840	0.825	1.000	1.000	1.000	1.000	0.504	-0.017	0.030	0.037	0.630	0.064	0.044	0.044	0.358	0.063	0.083	0.086	0.415	0.066	0.129	0.110

Table 3: Explainability quality of ExCut compared to the baselines

	Methods	UWCSE			IMDB			Hepatitis			Mutagenesis			WebKB			Terrorist		
		Cov	Exc	WRA	Cov	Exc	WRA	Cov	Exc	WRA	Cov	Exc	WRA	Cov	Exc	WRA	Cov	Exc	WRA
SOA	ReCeNT	0.914	0.879	0.136	1.000	0.038	0.009	1.000	0.001	0.000	1.000	0.000	0.000	1.000	1.000	0.000	0.930	0.424	0.058
	DEC	0.728	0.314	0.070	1.000	0.025	0.006	1.000	0.005	0.001	1.000	0.000	0.000	1.000	0.057	0.014	0.598	0.129	0.020
ExCut-T Kmeans	Kmeans-T	0.832	0.756	0.157	0.735	0.113	0.009	0.810	0.087	0.021	0.754	0.108	0.026	0.754	0.108	0.026	0.493	0.173	0.023
	belongToCl	0.888	0.888	0.189	1.000	1.000	0.105	0.755	0.637	0.134	0.94	0.388	0.094	0.984	0.116	0.012	0.680	0.259	0.029
	sameClAs	0.896	0.896	0.187	1.000	1.000	0.105	0.939	0.446	0.088	0.962	0.500	0.124	0.992	0.044	0.005	0.874	0.493	0.062
	entExplCl	0.896	0.896	0.187	1.000	1.000	0.105	0.754	0.636	0.134	0.994	0.477	0.115	0.991	0.095	0.014	0.943	0.799	0.107
	followExpl	0.896	0.896	0.187	1.000	1.000	0.105	0.749	0.631	0.134	0.975	0.456	0.111	0.989	0.092	0.014	0.952	0.790	0.114
ExCut-C Kmeans	Kmeans-C	0.592	0.059	0.007	0.725	0.038	0.005	0.613	0.085	0.021	0.869	0.302	0.075	0.984	0.040	0.007	0.645	0.279	0.020
	belongToCl	0.882	0.856	0.183	1.000	1.000	0.105	0.730	0.499	0.115	0.874	0.308	0.077	0.976	0.075	0.010	0.676	0.315	0.024
	sameClAs	0.910	0.892	0.187	1.000	1.000	0.105	0.802	0.449	0.112	0.870	0.303	0.076	0.982	0.096	0.010	0.849	0.606	0.067
	entExplCl	0.878	0.878	0.194	0.732	0.177	0.013	0.851	0.727	0.179	0.874	0.308	0.077	0.973	0.079	0.010	0.682	0.334	0.028
	followExpl	0.897	0.891	0.188	1.000	1.000	0.105	0.810	0.660	0.122	0.872	0.306	0.076	0.974	0.069	0.010	0.673	0.299	0.026
ExCut-T Spectral	Spectral-T	0.863	0.841	0.189	0.882	0.118	0.000	0.802	0.078	0.019	0.755	0.121	0.030	0.979	0.061	0.013	0.457	0.110	0.013
	belongToCl	0.877	0.877	0.194	1.000	1.000	0.105	0.856	0.739	0.183	0.949	0.419	0.103	0.997	0.099	0.019	0.798	0.633	0.079
	entExplCl	0.877	0.877	0.194	1.000	1.000	0.105	0.823	0.676	0.166	0.959	0.429	0.105	0.988	0.093	0.016	0.912	0.713	0.088
	entExplCl	0.877	0.877	0.194	1.000	1.000	0.105	0.857	0.752	0.178	0.894	0.344	0.086	0.991	0.086	0.013	0.746	0.403	0.057
ExCut-T DBSCAN	DBSCAN-T	0.670	0.670	0.000	1.000	1.000	0.000	1.000	1.000	0.000	0.714	0.138	0.004	1.000	1.000	0.000	0.771	0.640	0.011
	belongToCl	0.762	0.517	0.065	0.950	0.901	0.020	1.000	1.000	0.000	1.000	1.000	0.000	1.000	1.000	0.000	0.776	0.724	0.032
	entExplCl	0.867	0.688	0.134	1.000	1.000	0.000	1.000	1.000	0.000	1.000	1.000	0.000	1.000	1.000	0.000	0.907	0.832	0.020
	followExpl	0.962	0.788	0.137	0.952	0.920	0.023	1.000	1.000	0.000	1.000	1.000	0.000	1.000	1.000	0.000	0.777	0.745	0.035
ExCut-T Heir.	Heir-T	0.716	0.189	0.032	0.499	0.140	0.029	0.533	0.128	0.027	0.779	0.116	0.028	0.972	0.103	0.012	0.456	0.132	0.014
	belongToCl	0.877	0.877	0.194	0.731	0.610	0.152	0.841	0.723	0.173	0.831	0.259	0.064	0.980	0.174	0.019	0.641	0.448	0.078
	entExplCl	0.877	0.877	0.194	0.635	0.633	0.080	0.848	0.739	0.177	1.000	0.941	0.032	1.000	0.255	0.020	0.891	0.702	0.098
	followExpl	0.879	0.879	0.193	1.000	1.000	0.105	0.805	0.540	0.067	0.876	0.376	0.093	0.963	0.183	0.009	1.000	0.877	0.147
	Ground truth	0.920	0.896	0.187	1.000	1.000	0.105	0.922	0.566	0.137	1.000	0.163	0.039	1.000	0.038	0.010	0.643	0.333	0.027

Table 4: Quality of the clusters and the explanations found in Large-scale KGs

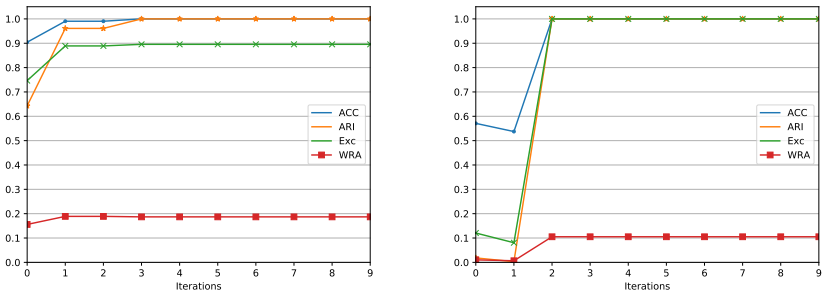
	Methods	LUBM Courses						Yago Artwork					
		ACC	ARI	NMI	Cov	Exc	WRA	ACC	ARI	NMI	Cov	Exc	WRA
Base.	DEC	0.917	0.696	0.656	0.963	0.952	0.189	0.555	0.435	0.567	0.921	0.488	0.111
	Kmeans-T	0.502	0.000	0.000	0.459	0.034	0.009	0.521	0.417	0.578	0.924	0.471	0.110
ExCut-T Kmeans	belongToCl	1.000	1.000	1.000	1.000	1.000	0.249	0.822	0.625	0.588	0.846	0.704	0.155
	sameClAs	0.876	0.567	0.517	0.910	0.793	0.185	0.968	0.908	0.896	0.952	0.930	0.207
	entExplCl	1.000	1.000	1.000	1.000	1.000	0.249	0.972	0.919	0.910	0.952	0.930	0.207
	followExpl	1.000	1.000	1.000	1.000	1.000	0.249	0.883	0.734	0.700	0.855	0.780	0.173
Ground Truth		-	-	-	1.000	1.000	0.249	-	-	-	0.953	0.928	0.206

Table 5: Explanations of clusters `song`, `book`, and `movie` from Yago KG.

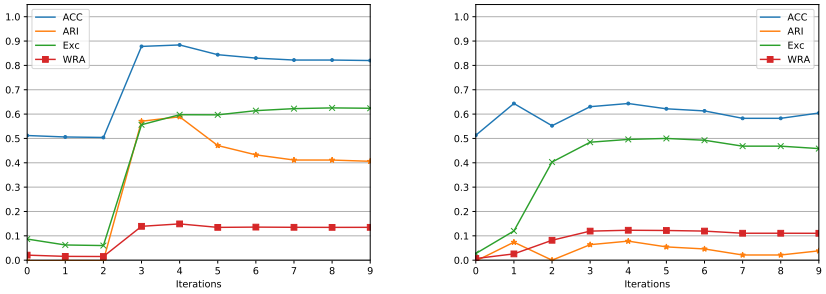
$X \in$	Kmeans-T				ExCut-T			
	Explanations	Cov	Exc	WRA	Explanations	Cov	Exc	WRA
C_1	$created(Y,X), bornIn(Y,Z)$	0.94	0.55	0.13	$created(Y,X), type(Y,\texttt{artist})$	0.99	0.96	0.21
	$created(Y,X), type(Y,\texttt{artist})$	0.49	0.45	0.10	$created(Y,X), won(Y,\texttt{grammy})$	0.57	0.57	0.12
	$created(Y,X), type(Y,\texttt{writer})$	0.52	0.44	0.10	$created(Y,X), type(Y,\texttt{living\_pers})$	0.84	0.48	0.11
C_2	$directed(Y,X)$	0.92	0.56	0.11	$created(Y,X), type(Y,\texttt{writer})$	0.99	0.91	0.19
	$directed(Y,X), gender(Y,\texttt{male})$	0.89	0.54	0.10	$created(Y,X), diedIn(Y,Z)$	0.46	0.20	0.04
	$created(Y,X), type(Y,\texttt{person})$	0.71	0.52	0.06	$created(Y,X)$	1.00	0.00	0.05
C_3	$actedIn(Y,X), type(Y,\texttt{living\_pers})$	0.58	0.30	0.07	$actedIn(Y,X)$	0.81	0.81	0.19
	$locatedIn(X,Y), hasLang(Y,Z)$	0.60	0.29	0.07	$actedIn(Y,X), bornIn(Y,Z)$	0.79	0.79	0.18
	$locatedIn(X,Y), currency(Y,Z)$	0.60	0.29	0.07	$actedIn(Y,X), type(Y,\texttt{person})$	0.78	0.78	0.18

Table 6: Comparing average Silhouette Width for ExCut configurations against the base baseline

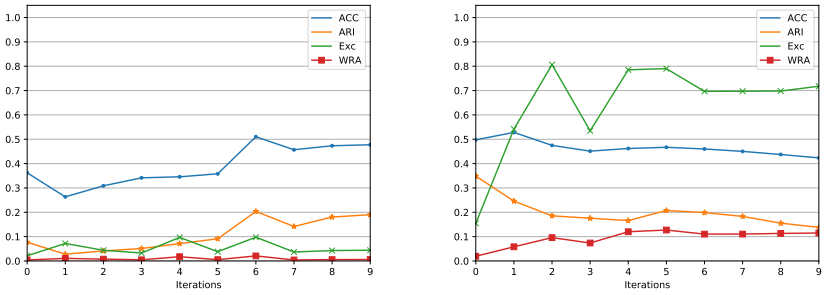
Method		UWCSE	IMDB	Hepatitis	Mutagenesis	WebKB	Terrorist	LUBM	Yago
ExCut-T Kmeans	Kmeans-T	0.076	0.016	0.078	0.249	0.186	0.063	0.817	0.570
	belongToCl	0.177	0.223	0.098	0.037	0.043	*0.046*	0.216	0.124
	sameClAs	0.438	0.334	0.246	0.257	0.324	0.487	0.146	0.285
	entExplCl	0.285	0.234	0.159	0.060	0.041	0.046	0.216	0.240
	followExpl	0.241	0.205	0.189	0.050	0.038	0.030	0.248	0.110
ExCut-C Kmeans	Kmeans-C	0.010	0.009	0.007	0.008	0.003	0.005	-	-
	belongToCl	0.150	0.169	0.105	0.050	0.043	0.070	-	-
	sameClAs	0.431	0.392	0.643	0.048	0.066	0.156	-	-
	entExplCl	0.150	0.114	0.122	0.051	0.042	0.068	-	-
	followExpl	0.241	0.129	0.217	0.050	0.046	0.065	-	-



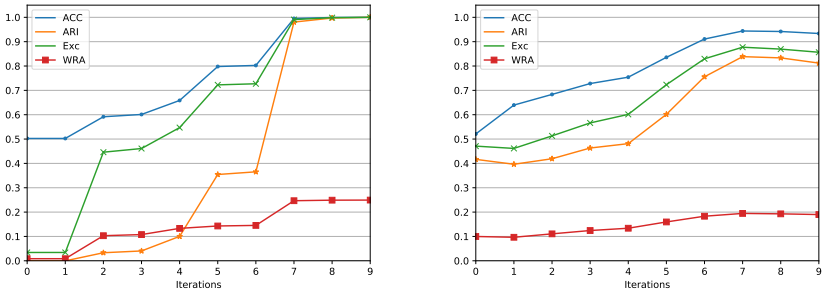
(a) UWCSE (b) IMDB



(c) Hepatitis (d) Mutagenesis



(e) Webkb (f) Terrorist



(g) LUBM (h) Yago Artwork

Fig. 5: Quality of resulting clusters and their explanations using ExCut-T

Table 7: Effect of including history and context

	Context	History	ACC	ARI	HMS	NMI	CCov	XCov	WRA
Hepatities	-	-	0.514	-0.002	0.000	0.000	0.808	0.084	0.020
	No	No	0.708	0.182	0.133	0.136	0.836	0.728	0.170
	No	Yes	0.828	0.429	0.328	0.336	0.757	0.692	0.157
	Yes	No	0.744	0.250	0.183	0.190	0.815	0.706	0.160
	Yes	Yes	0.736	0.240	0.183	0.191	0.818	0.708	0.161
IMDB	-	-	0.559	0.009	0.055	0.039	0.760	0.062	0.005
	No	No	0.946	0.825	0.817	0.812	0.940	0.924	0.110
	No	Yes	0.910	0.788	0.829	0.821	0.938	0.921	0.111
	Yes	No	0.946	0.825	0.817	0.812	0.940	0.924	0.110
	Yes	Yes	0.947	0.830	0.823	0.817	0.940	0.923	0.110
Mutagenesis	-	-	0.517	-0.004	0.003	0.003	0.754	0.108	0.026
	No	No	0.636	0.069	0.081	0.080	0.986	0.464	0.112
	No	Yes	0.622	0.054	0.062	0.061	1.000	0.500	0.122
	Yes	No	0.667	0.109	0.120	0.120	0.935	0.374	0.089
	Yes	Yes	0.635	0.068	0.077	0.076	0.993	0.482	0.117
Terrorist	-	-	0.511	0.350	0.601	0.482	0.503	0.100	0.020
	No	No	0.449	0.159	0.267	0.217	0.790	0.442	0.085
	No	Yes	0.379	0.117	0.213	0.172	0.863	0.300	0.079
	Yes	No	0.495	0.224	0.318	0.261	0.735	0.316	0.043
	Yes	Yes	0.406	0.127	0.218	0.176	0.772	0.430	0.091
UWSCE	-	-	0.957	0.831	0.728	0.721	0.901	0.867	0.186
	No	No	0.989	0.954	0.936	0.930	0.889	0.889	0.189
	No	Yes	1.000	1.000	1.000	1.000	0.896	0.896	0.187
	Yes	No	1.000	1.000	1.000	1.000	0.896	0.896	0.187
	Yes	Yes	1.000	1.000	1.000	1.000	0.896	0.896	0.187
WebKB	-	-	0.367	0.078	0.100	0.093	0.983	0.009	0.010
	No	No	0.379	0.093	0.129	0.121	0.998	0.025	0.011
	No	Yes	0.320	0.024	0.067	0.062	0.998	0.019	0.015
	Yes	No	0.387	0.098	0.145	0.137	0.989	0.024	0.010
	Yes	Yes	0.368	0.085	0.158	0.146	0.995	0.024	0.011

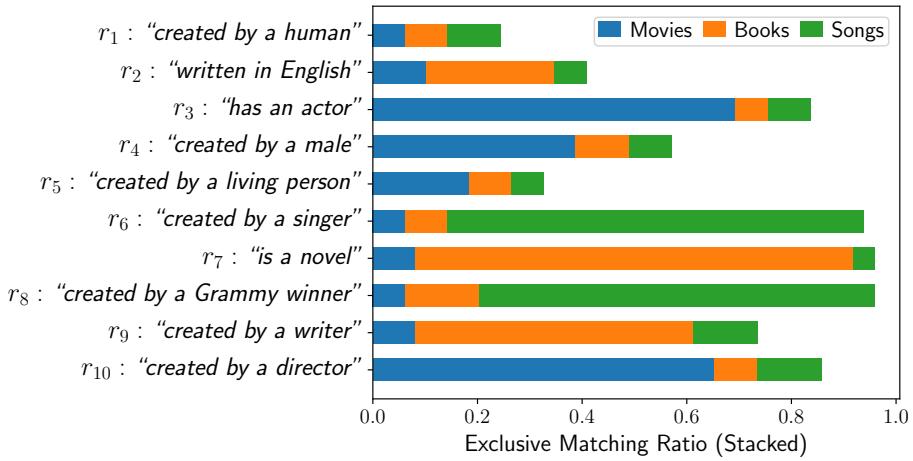


Fig. 6: Ratio of explanation-to-cluster pairs exclusively matched.

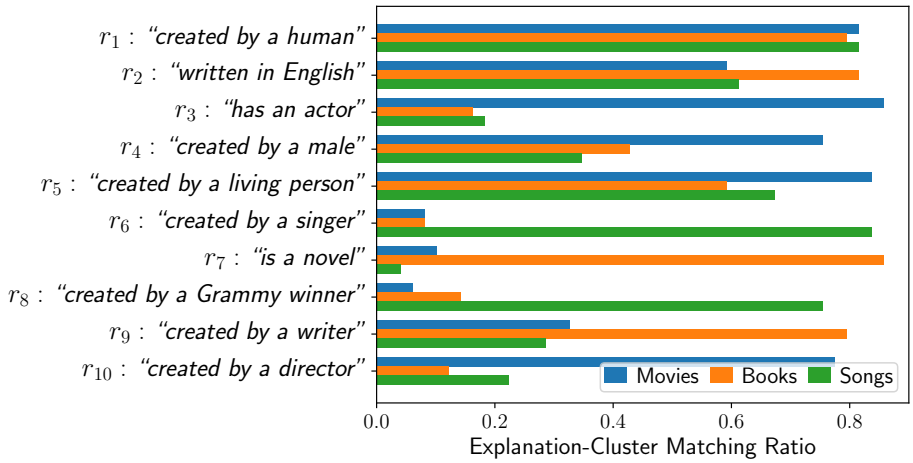


Fig. 7: Ratio of explanation-cluster matches pairs to the total number of valid responses