User Scored Evaluation of Non-Unique Explanations for Relational Graph Convolutional Network Link Prediction on Knowledge Graphs

Nicholas Halliwell, Fabien Gandon, Freddy Lecue

User Scored Evaluation of Non-Unique Explanations for Relational Graph Convolutional Network Link Prediction on Knowledge Graphs

Nicholas Halliwell
Inria, UCA, CNRS, I3S
Sophia Antipolis, France
nicholas.halliwell@inria.fr

Fabien Gandon
Inria, UCA, CNRS, I3S
Sophia Antipolis, France
fabien.gandon@inria.fr

Freddy Lecue
CortAIx, Thales
Montreal, Canada
freddy.lecue@thalesgroup.com

ABSTRACT

Relational Graph Convolutional Networks (RGCNs) are commonly used on Knowledge Graphs (KGs) to perform black box link prediction. Several algorithms, or explanation methods, have been proposed to explain their predictions. Evaluating performance of explanation methods for link prediction is difficult without ground truth explanations. Furthermore, there can be multiple explanations for a given prediction in a KG. No dataset exists where observations have multiple ground truth explanations to compare against. Additionally, no standard scoring metrics exist to compare predicted explanations against multiple ground truth explanations. In this paper, we introduce a method, including a dataset (FrenchRoyalty-200k), to benchmark explanation methods on the task of link prediction on KGs, when there are multiple explanations to consider. We conduct a user experiment, where users score each possible ground truth explanation based on their understanding of the explanation. We propose the use of several scoring metrics, using relevance weights derived from user scores for each predicted explanation. Lastly, we benchmark this dataset on state-of-the-art explanation methods for link prediction using the proposed scoring metrics.

CCS CONCEPTS

• **Mathematics of computing → Computing most probable explanation**; • **Computing methodologies → Knowledge representation and reasoning**; *Neural networks.*

KEYWORDS

link prediction; Explainable AI; knowledge graphs; graph neural networks; explanation evaluation

ACM Reference Format:
Nicholas Halliwell, Fabien Gandon, and Freddy Lecue. 2021. User Scored Evaluation of Non-Unique Explanations for Relational Graph Convolutional Network Link Prediction on Knowledge Graphs. In *Proceedings of ACM Conference (K-CAP 2021).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3460210.3493557

1 INTRODUCTION

Knowledge Graphs represent facts as triples in the form *(subject, predicate, object)*, where a *subject* and *object* represent a real-world entity, linked by some *predicate*. Knowledge Graphs often do not explicitly contain every available fact. Link prediction on Knowledge Graphs is used to identify unknown facts from existing ones. One approach to link prediction on Knowledge Graphs involves the use of graph embedding algorithms that learn a function mapping each subject, object, and predicate to a low-dimensional space. A scoring function is defined to quantify if a link (relation) exists between two nodes (entities). Relational Graph Convolutional Networks (RGCN) [9] extends Graph Convolutional Networks [7] for applications to link prediction on Knowledge Graphs, using the scoring function from DistMult [11] as an output layer, returning a probability of the input triple being a fact.

The decision function of a black box link predictor such as an RGCN gives no insight, or explanation, as to why the model arrives at a particular decision. As a result, several algorithms for explainable link prediction have been proposed, in particular: ExplaiNE [5] quantifies how the predicted probability of a link changes when weakening or removing a link with a neighboring node, while GNNExplainer [12] explains the predictions of any Graph Neural Network, learning a mask over the adjacency matrix to identify the most informative subgraph. Explanations from ExplaiNE and GNNExplainer return explanations to the user in the form of existing triples in the Knowledge Graph.

Ground truth explanations however can be non-unique. There can be multiple, logically correct ways to explain why a link could exist between two nodes. Consider an example where a model predicts the *hasChild* link between two entities Louis VII of France, and Agnes of France, i.e. (Louis VII, *hasChild*, Agnes of France) . One way to explain why this link could exist between these two entities is because Agnes of France is the child of Louis VII's spouse Adela of Champagne. This is not the only way to explain why the *hasChild* link exists between Louis VII and Agnes. Agnes could be the child of Louis VII because Agnes' grandparent, Louis VI is the parent of Louis VII. Both of these explanations are correct, it is unclear as to which explanation is optimal.

State-of-the-art explanation methods for link prediction on Knowledge Graphs have no common dataset or performance metrics to quantitatively evaluate explanation quality when there are multiple ways to explain the model's prediction. In this paper, we propose a method, including a dataset (FrenchRoyalty-200k), to quantitatively evaluate explanation methods on the task of link prediction using Graph Neural Networks, when there are multiple explanations. This dataset includes all possible ground truth explanations for each triple, allowing for quantitative comparisons across every possible explanation. Additionally, we perform a user experiment, where users decide which explanations are optimal. Furthermore, we propose the use of several scoring metrics using these user scores as relevance weights for each predicted explanation. Lastly,

K-CAP 2021, December 2021, Online
2021. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/3460210.3493557

we benchmark this dataset on state-of-the-art explanation methods using the proposed dataset and evaluation metrics.

2 RELATED WORK, SHORTCOMINGS, AND CONTRIBUTIONS

Knowledge Graph embeddings. Knowledge Graph embedding algorithms learn continuous vectors for each subject, predicate and object. A scoring function assigns a value to each triple based on if the subject, predicate, and object form a valid fact. TransE [1] models relationships as translation operations on learned entity vectors. DistMult [11] learns a diagonal matrix of parameters for each relation. We refer the reader to a recent survey [4] for more information. A Relational Graph Convolutional Networks (RGCN) [9] can be used to learn embeddings and perform link prediction on Knowledge Graphs. The RGCN performs embedding updates for a given entity by multiplying the neighboring entities with a weight matrix for each relation in the dataset, and summing across each neighbor and relation. There are many approaches for link prediction (e.g. rule based, bayesian, etc.), however, this work focuses on the evaluation and explanation of link prediction on Knowledge Graphs using Graph Neural Networks.

Explainable link prediction. Few algorithms exist to understand the predictions of Knowledge Graph embedding algorithms. For some model with scoring function g, ExplaiNE [5] computes the gradient of the scoring function with respect to the adjacency matrix. This measures the change in score due to a small perturbation in the adjacency matrix, that is, how much will the score change if a link is added or removed between two given nodes. Formally, given two nodes i, j serving as prediction candidates, and two nodes k, l serving as a candidate explanation, the score assigned to node pair k, l is given by Equation 1, where $\mathbf{X}^*$ is the optimal embedding matrix, and a_{kl} is an element of the adjacency matrix $\mathbf{A}$.

$$\frac{\partial g_{ij}}{\partial a_{kl}}(\mathbf{A}) = \nabla_{\mathbf{X}} g_{ij}(\mathbf{X}^*)^T \cdot \frac{\partial \mathbf{X}^*}{\partial a_{kl}}(\mathbf{A}) \tag{1}$$

GNNExplainer [12] explains the predictions of any Graph Neural Network, learning a mask over the input adjacency matrix to identify the most relevant subgraph. This is achieved by minimizing the cross entropy between the predicted label using the input adjacency matrix, and the predicted label using the masked adjacency matrix. The objective function minimized by GNNExplainer is given by Equation 2, where $\mathbf{M}$ is a mask learned and $\odot$ denotes element-wise multiplication.

$$\min_{\mathbf{M}} - \sum_{c=1}^{C} \mathbb{1}[y = c] \, log P_{\Phi}(Y = y | \mathbf{A}_c \odot \sigma(\mathbf{M}), \mathbf{X}_c) \tag{2}$$

Explanation quality. The weakness of these explanation methods is the empirical evaluation of explanation quality. The authors of ExplaiNE acknowledge the difficulty in measuring the quality of explanation due to the lack of available datasets with ground truth explanations [5]. ExplaiNE relies on the assumption that an explanation can be found using one of the 1^{st} degree neighbors. On the task of movie recommendation, ExplaiNE measures the quality of explanations using the average Jaccard similarity between the genres for a given recommended movie, and the set of genres from the top 5 ranked explanations computed. A p-value is then computed to estimate the significance of the average. It is unknown how this evaluation method generalizes to tasks outside of movie recommendation. ExplaiNE has been previously benchmarked with 4 datasets: Karate, DBLP, MovieLens, and Game of Thrones networks. These datasets do not include ground truth explanations, and defining ground truth explanations for these networks is non-trivial. GNNExplainer has not been benchmarked on the task of explainable link prediction on Knowledge Graphs due to the lack of available datasets. Both GNNExplainer and ExplaiNE lack a common dataset and metric to evaluate explanation quality.

Multiple ground truths. There can be multiple ways to explain why a link could exist between two nodes. Not all explanations are equally informative about the model's decision, some explanations can be arbitrarily more complicated than others. Explanation methods could generate an explanation when a more intuitive explanation could exist. Datasets containing only one ground truth explanation for each observation are insufficient to quantitatively evaluate explanation methods. Without considering all possible explanations, an explanation method could be incorrectly penalized for identifying a correct explanation not included in the ground truths. Therefore, a predicted explanation must be evaluated against all possible ground truth explanations. To our knowledge, there exists no dataset for link prediction on Knowledge Graphs where all possible ground truth explanations are included. Additionally, there are no standard quantitative metrics to measure the quality of explanations generated when there are multiple explanations to consider, making quantitative comparisons across explanations difficult.

Contributions. Our contributions include a method to quantitatively evaluate explanation methods on the task of link prediction on Knowledge Graphs, when there are multiple ground truth explanations to consider. Additionally, we propose a dataset, FrenchRoyalty-200k, that includes every possible ground truth explanation for each observation. We perform a user experiment to determine which ground truth explanations are most intuitive. Furthermore, we propose the use of several performance metrics that score predicted explanations based on how intuitive users found the explanation, allowing for quantitative comparisons across explanation methods. Lastly, we benchmark state-of-the-art explanation methods using the proposed dataset and metrics.

3 GENERATING A USER EVALUATED DATASET WITH GROUND TRUTH EXPLANATIONS

3.1 Inference Traces as Explanations

In a Knowledge Graph, the available formal semantics allow us to view ground truth explanations as equivalent to computing justification for an entailment. We select an open-source semantic reasoner with rule-tracing capabilities [2] to generate ground truth explanations for each defined rule, without needing manual annotations. This tracing pinpoints the input triples that caused the generation of a triple we will then try to predict and explain.

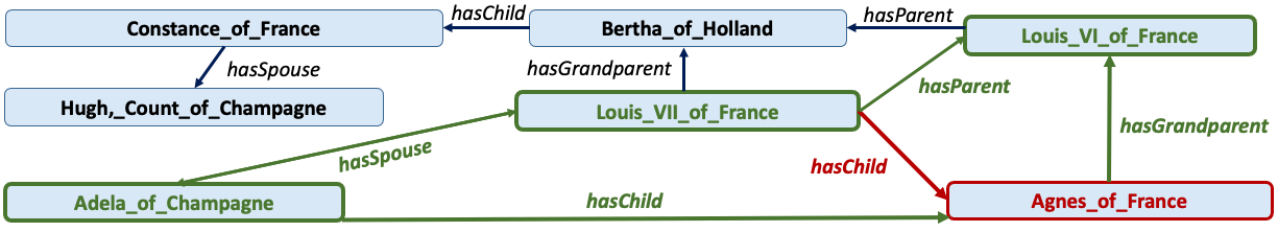

Figure 1: A candidate triple (*Louis VII of France, hasChild, Agnes of France*) plotted in red with its non-unique explanations in green: {(*Agnes of France, hasGrandparent, Louis VI of France*), (*Louis VII of France, hasParent, Louis VI of France*)}, {(*Adela of Champagne, hasChild, Agnes of France*), (*Louis VII of France, hasSpouse, Adela of Champagne*)}, and neighboring triples.

We rely on a set of rules equivalent to strict Horns clauses i.e. disjunctions of literals with exactly one positive literal l_c, all the other l_i being negated: $\neg l_1 \vee ... \vee \neg l_n \vee l_c$. The implication form of the clause can be seen as an inference rule assuming that, if all l_i hold (the antecedent of the rule), then the consequent l_c also holds, denoted $l_c \leftarrow l_1 \wedge ... \wedge l_n$. Each literal is a binary predicate capturing a triple pattern of the Knowledge Graph with variables universally quantified for the whole clause. For instance, $hasGrandparent(X, Z) \leftarrow hasParent(X, Y) \wedge hasParent(Y, Z)$.

For a given Knowledge Graph and a given set of rules, the semantic reasoner performs a forward chaining materialization of all inferences that can be made. Each time the engine finds a mapping of triples $T_1, \ldots, T_n$ making the antecedent of a rule true, it materializes the consequent triple T_c, and records the explanations in the form $T_c \leftarrow (T_1, \ldots, T_n)$, where T_c is a generated triple, and triples $T_1, \ldots, T_n$ are its explanation.

Indeed this forms an intuitive explanation for graph data: a study shows users prefer example based explanations [3] and non-personalized feature-based explanations are efficient [10]. This generic approach to generating ground truth explanations can be applied to many Knowledge Graphs and many sets of rules. In this work, we focus on non-unique explanations, i.e. logical rules constructed to include all possible ground truth explanations. To our knowledge, this approach to generating non-unique ground truth explanations has not been applied to the task of explainable link prediction on Knowledge Graphs using Graph Neural Networks. All the resources used and produced in this work are available online including the download link for the reasoner, code for this paper and datasets[1].

3.2 Ensuring Completeness of Explanations

In this paper, we focus on providing a dataset with non-unique explanations. We chose to describe family relations as no prior domain knowledge is needed. The explanation methods we want to evaluate provide their explanations as a set of triples that justify a prediction. In order to construct a dataset that includes all possible explanations for a given predicted triple, we first enumerated all possible paths between the two nodes involved in this predicted triple. To exhaustively list all possible cases, we defined a small synthetic family graph systematically using all the possible types of family relations. This graph describes some individual Paul, and all family members within a 2-hop neighborhood, this includes aunts, grandparents, children, etc. This complete synthetic graph is then used to identify all possible paths with a maximum length of 2 linking the subject and objects of its triples. This graph is purposely kept small, to ensure each possible path can be verified manually. Each of these paths corresponds to one possible explanation as to why a link could exist between two given nodes e.g., Paul and Tom are brothers because Paul and Tom both have the same parent Jim. Indeed some of these paths can be turned into the antecedent of an inference rule for that type of triple e.g. $hasGrandparent(X, Z) \leftarrow hasParent(X, Y) \wedge hasParent(Y, Z)$. We define paths that can always be turned into the antecedent of an inference rule as *logical explanations*. In other words, these explanations are always true. Other paths do not always logically imply the targeted triple but still provide a good indication that could have triggered a human guess or a statistical prediction. For instance, $hasParent(X, Y) \wedge hasParent(Z, Y)$ could indicate X and Z are brothers or sisters or any combination of these relations. Without additional knowledge (e.g. the gender) the explanation is not always logically true. We define these paths as *partial explanations*. Some explanations will be more convincing than others to a user, especially among partial explanations. A score is needed for each possible explanation to distinguish good, intuitive explanations from bad, unintuitive ones.

3.3 Logical Derivation and Partial Explanation Rules

In this work, we focus on 6 family relationships: *hasSpouse*, with 3 possible explanations, *hasBrother*, with 7 possible explanations, *hasSister*, with 7 possible explanations, *hasGrandparent*, with 6 possible explanations, and *hasChild*, and *hasParent* with 9 possible explanations. As we have two types of explanations (logical vs. partial), there are also have two types of rules (logical derivation vs. partial explanation). We define a *logical derivation rule* as one that is always true, and a *partial explanation rule* as one that is not always true without additional information, such as gender. The predicate of each triple used on the link prediction task is in the consequent of one or more of these rules. The associated explanation consists of the all possible triples that triggered each rule. The logical derivation rules trigger every time their antecedent is matched, and its corresponding triple and logically true explanation are generated. The partial explanation rules trigger only if the triple is already known (asserted or inferred by other rules) and are just adding alternative partial explanations, therefore preventing any

[1]https://github.com/halliwelln/multiple-explanations/

false triples from being included in the graph. In addition, each rule (both logical derivation and partial explanation) associates a score to the explanations it generates. The score is defined at the per rule level, and is the same for all the explanations generated by that rule. The score captures how intuitive a given pattern of explanation is for a given type of predicted triple, detailed in the next section.

We apply all rules to the entire Knowledge Graph of the French Royalty families found in DBpedia [8] to build the FrenchRoyalty-200k dataset. Included with each triple in the training and test sets are all possible explanations as to why a given link could exist between the two nodes. Figure 1 shows an example of a candidate triple along with several possible explanations. Table 1 outlines all rules defined in the FrenchRoyalty-200k dataset. Included in this table is the number of total triples for each predicate, the predicates used to define every possible explanation, the user score assigned to each predicate, and a column to indicate whether a rule is logically true or provides a partial explanation.

3.4 Users' Evaluation of Explanation Scores

Although ExplaiNE and GNNExplainer have many possible explanations to choose from, these explanations are not equal. Some explanations may be easier to understand than others. When benchmarking explanation methods with non-unique explanations, a scoring metric should assign a high score when an explanation method correctly predicts an intuitive explanation, and a low score when an unintuitive, overly complicated explanation is predicted.

We conducted a user experiment to score each possible explanation. This allows us to distinguish explanations that are intuitive from those that are not without relying on any prior assumptions. One could rely on the assumption that for example the shortest path, i.e., the explanation that uses the fewest number of predicates, is the most intuitive explanation. This assumption would fail when predicting the *hasGrandparent* predicate, as there are no 1 hop paths, but many 2 hop paths. Relying on the shortest path would treat all 2 hop paths as equally intuitive, while the *hasParent/hasParent* path is by far the best explanation for *hasGrandparent*.

Using a survey platform, we conduct an experimental evaluation where for each predicate, users are shown all possible explanations on the Paul's Family graph, and asked to assign a score to each path based on if the explanation is intuitive. Users are given the following definition: "An explanation is considered intuitive if it is easily and immediately understood." For each predicate, and for each possible explanation, users are given an example of the predicate and its explanation used in a sentence. For example, for the *hasSister* predicate, one explanation uses the *hasParent*, and *hasChild* relations. Users are asked to score the following explanation: "Ruth is the *sister* of Paul because Mary is the *parent* of Paul, and Ruth is the *child* of Mary."

Users scored each of these explanations on a five-point Likert scale: (4) Very intuitive, an explanation I could give or expect; (3) Intuitive; (2) Neither intuitive or unintuitive; (1) Unintuitive ; (0) Not intuitive at all, not an explanation I would give or expect.

In total, 42 users responded from 11 different nationalities, consisting of both computer science and non-computer science backgrounds. We normalized the average scores between 0 and 1 for each possible explanation, and round them to the nearest tenth. These user scores are used in the rules and in the benchmark, detailed below, to penalize unintuitive predicted explanations, and reward intuitive predicted explanations. User scores for each predicate and explanation can be found in Table 1.

This survey also showed that, even for humans, explanations can be difficult to define and assess. Users had difficulty deciding which explanations were intuitive. Even when presented equivalent explanations for two different predicates, for example, using *hasBrother*, *hasSister* to explain a *hasBrother* link, and using *hasBrother*, *hasSister* to explain a *hasSister* link, users on average did not assign these explanations the same score. This lack of symmetry can be seen for multiple predicates in Table 1.

4 EVALUATION OF NON-UNIQUE EXPLANATIONS

4.1 Scoring Metrics

To our knowledge, there is no standard evaluation metric to measure the quality of explanations generated by explanation methods when there are non-unique explanations available to predict. A standard evaluation metric is needed to identify when one explanation method is preferable to the other. The binary precision and recall could be used for this task, however, these metrics fail to account for the fact that some explanations can be more intuitive than others to users. Both metrics would give a score of 1 when a predicted explanation exactly matches a ground truth explanation. However, an explanation method could predict an unintuitive explanation, and receive the highest possible evaluation score, potentially misleading practitioners into thinking the predicted explanation is of high quality. Therefore, scoring metrics used for this task must compare a predicted explanation to all possible explanations, and account for the fact that explanations have different degrees of relevance. Ideally, a scoring metric for this task should assign a lower score to an unintuitive predicted explanation, and a higher score to an intuitive predicted explanation.

We propose to score explanation methods with non-unique explanations by adapting the generalized precision and generalized recall [6]. Originally proposed for document retrieval, generalized precision and generalized recall measure precision and recall based on the relevance score assigned to each retrieved document. Generalized precision is defined by the sum of relevance scores for each retrieved document divided by the number of retrieved documents. Generalized recall is defined by the sum of relevance scores for each retrieved document divided by the sum of relevance scores for all documents in the database.

We adapted these metrics in the context of link prediction on Knowledge Graphs. Formally, let t_i be a triple, $e_i = \{t_1, \ldots, t_n\}$ be one of the possible ground truth explanations for triple t_i. Let $\hat{e}_i \in E_i$ be the predicted explanation for t_i, and E_i be defined as all possible explanations for t_i. Lastly, let $s(.)$ gives the relevance score (determined by the user experiment) for a given explanation. First, the best possible user score for an explanation is given by Equation 3:

$$s_i = \max_{e_i \in E_i} s(e_i) \tag{3}$$

Predicate	# Triples	Explanations	User Score	Explanation Type	Predicate	# Triples	Explanations	User Score	Explanation Type
hasBrother	6, 067	*hasSister*	0.8	Partial	hasSister	4,433	*hasBrother*	0.8	Partial
		hasChild, *hasParent*	0.8	Partial			*hasChild*, *hasParent*	0.8	Partial
		hasGrandparent (x2)	0.3	Partial			*hasGrandparent* (x2)	0.2	Partial
		hasBrother, *hasSister*	0.6	Logical			*hasBrother*, *hasSister*	0.7	Logical
		hasBrother, *hasBrother*	0.7	Logical			*hasBrother*, *hasBrother*	0.7	Partial
		hasSister, *hasSister*	0.7	Partial			*hasSister*, *hasSister*	0.7	Logical
		hasParent, *hasParent*	0.9	Partial			*hasParent*, *hasParent*	0.9	Partial
hasChild	52,399	*hasParent*	0.9	Logical	hasParent	48,241	*hasChild*	0.9	Logical
		hasChild, *hasSister*	0.7	Logical			*hasChild*, *hasSister*	0.7	Logical
		hasBrother, *hasChild*	0.7	Logical			*hasBrother*, *hasChild*	0.7	Logical
		hasChild, *hasSpouse*	0.7	Logical			*hasChild*, *hasSpouse*	0.7	Logical
		hasGrandparent, *hasParent*	0.4	Partial			*hasGrandparent*, *hasParent*	0.3	Partial
		hasChild, *hasGrandparent*	0.4	Partial			*hasChild*, *hasGrandparent*	0.3	Partial
		hasBrother, *hasParent*	0.7	Logical			*hasBrother*, *hasParent*	0.7	Logical
		hasParent, *hasSpouse*	0.7	Logical			*hasParent*, *hasSpouse*	0.7	Logical
		hasParent, *hasSister*	0.7	Logical			*hasParent*, *hasSister*	0.7	Logical
hasSpouse	31,984	*hasSpouse*	0.8	Logical	hasGrandparent	61,333	*hasGrandparent*, *hasSister*	0.6	Logical
		hasChild,hasParent	0.5	Logical			*hasChild*, *hasParent*	0.9	Logical
							hasBrother, *hasGrandparent*	0.6	Logical
							hasParent,hasParent	0.9	Logical
		hasChild,hasChild	0.9	Logical			*hasGrandparent*, *hasSpouse*	0.7	Logical
							hasChild, *hasChild*	0.9	Logical

Table 1: FrenchRoyalty-200k dataset: Breakdown of all predicates each possible explanation, and its score given by users. # Triples column denotes the total number of triples with that predicate. User Score column gives the score assigned to each explanation by users. Explanation Type column denotes whether this explanation is a logical (always true) or only partial.

The generalized precision between a predicted explanation and a ground truth explanation is given by Equation 4:

$$gp(\hat{e}_i, e_i) = \frac{|\hat{e}_i \cap e_i| \times s(e_i)}{|\hat{e}_i| \times s_i} \tag{4}$$

Intuitively, it is a sum of the user scores of the triples shared by the prediction and the ground truth, divided by the number of triples in the prediction, and the largest possible user score. For a given triple t_i, we take the highest generalized precision across all of t_i's ground truth explanations, given by Equation 5:

$$gp(\hat{e}_i, E_i) = \max_{e_i \in E_i} gp(\hat{e}_i, e_i) \tag{5}$$

We compute the maximum generalized precision for each triple, and average across the dataset. For the set of all predicted explanations $\hat{E}$, and the set of all ground truth explanation sets E, the generalized precision for an explanation method across the entire dataset is given by Equation 6:

$$GP(\hat{E}, E) = \frac{\sum_{\hat{e}_i \in \hat{E}, E_i \in E} gp(\hat{e}_i, E_i)}{|\hat{E}|} \tag{6}$$

The generalized recall sums the relevance scores for each predicted explanation that exists in the ground truth explanations, divided by the number of triples in the ground truth explanation, and the largest possible user score, given by Equation 7:

$$gr(\hat{e}_i, e_i) = \frac{|\hat{e}_i \cap e_i| \times s(e_i)}{|e_i| \times s_i}. \tag{7}$$

Similar to generalized precision, we propose to compute a maximum generalized recall for each triple in the dataset (Equation 8) and average it across the dataset (Equation 9).

$$gr(\hat{e}_i, E_i) = \max_{e_i \in E_i} gr(\hat{e}_i, e_i) \tag{8}$$

$$GR(\hat{E}, E) = \frac{\sum_{\hat{e}_i \in \hat{E}, E_i \in E} gr(\hat{e}_i, E_i)}{|\hat{E}|} \tag{9}$$

Note that we normalize the generalized precision and recall by the largest possible user score for each explanation to ensure they take values between 0 and 1.

We compute the generalized F_1 score, defined as the harmonic mean between the generalized precision and generalized recall. To ensure the recall and precision are computed on the same explanation, we compute them before we select the maximum (Equation 10) and average it (Equation 11)

$$gf_1(\hat{e_i}, E_i) = \max_{e_i \in E_i} \frac{2 \times gr(\hat{e_i}, e_i) \times gp(\hat{e_i}, e_i)}{gr(\hat{e_i}, e_i) + gp(\hat{e_i}, e_i)} \tag{10}$$

$$GF_1(\hat{E}, E) = \frac{\sum_{\hat{e_i} \in \hat{E}, E_i \in E} gf_1(\hat{e_i}, E_i)}{|\hat{E}|} \tag{11}$$

Finally we propose the use of the max-Jaccard metric to identify which explanation had the most in common with the predicted explanation. Formally, for triple t_i, we compute the Jaccard similarity between predicted explanation $\hat{e_i}$ with one of the possible ground truth explanation sets e_i, given by Equation 12:

$$j(e_i, \hat{e_i}) = \frac{|e_i \cap \hat{e_i}|}{|e_i \cup \hat{e_i}|} = \frac{|e_i \cap \hat{e_i}|}{|e_i| + |\hat{e_i}| - |e_i \cap \hat{e_i}|}. \tag{12}$$

We compute this Jaccard similarity across all possible explanations in set E_i for triple t_i (Equation 13) and average the result over the dataset (Equation 14):

$$mj(\hat{e_i}, E_i) = \max_{e_i \in E_i} j(\hat{e_i}, e_i) \tag{13}$$

$$MJ(\hat{E}, E) = \frac{\sum_{\hat{e_i} \in \hat{E}, E_i \in E} mj(\hat{e_i}, E_i)}{|\hat{E}|} \tag{14}$$

The max-Jaccard compares a predicted explanation with all possible explanations available to choose from. Intuitively it identifies the ground truth explanation that shares a maximum number of triples with the predicted explanation, therefore indicating which explanation a method may have tried to predict.

We argue these metrics are sufficient to quantitatively compare explanation methods when there are multiple explanations to consider. The max-Jaccard score measures if the explanation method is able to accurately predict one of the possible explanations to choose from. The generalized precision and generalized recall measure if the predicted explanation was given a high intuitive score assigned by users. Both metrics prevent an explanation method from only predicting low scored, unintuitive explanations, and receiving a high score. Lastly, the generalized F_1 provides an overview of performance on the generalized precision and recall.

4.2 Benchmark Setup and Protocol

One way to perform link prediction on Knowledge Graphs is to learn a real-valued vector for each entity and relation. For this benchmark, we use a Relational Graph Convolutional Network (RGCN) to learn embeddings. This was chosen as it can be used with many explanation methods without the need for any further adaptations. GNNExplainer is only defined for Graph Neural Networks, hence a GNN must be used on the link prediction task. ExplaiNE requires a model that takes an adjacency matrix as input. The RGCN meets both of these requirements. Additionally, the scoring function has a meaningful interpretation, returning the probability that an input triple is a fact. We fix the number of dimensions to 10, the best performing in terms of accuracy from the set $\{3, 5, 10\}$. We use a learning rate of 0.01, the best performing from the set $\{0.01, 0.001, 0.0001\}$. Lastly, we train the RGCN on 1000 epochs for all rules, found to the best performing from the set $\{50, 100, 500, 1000, 2000\}$. We report the accuracy of the RGCN on the link prediction task. For each data subset and each explanation method, we report the generalized precision, generalized recall, generalized F_1, and max-Jaccard.

We train GNNExplainer using a learning rate of 0.001 for each rule. We use 20 iterations for each observation. 3-fold cross validation is performed for both explanation methods, and we report the results of the best performing fold.

4.3 Results and Discussion

Results per Subset. We benchmark the FrenchRoyalty-200k dataset by splitting the full data into subsets where only one type of predictable predicate is included. The top half of Table 2 reports performance results of each predicate subset. For example, the Spouse subset included only triples in the training and test sets with the *hasSpouse* predicates, and their associated explanations.

First, the topmost row of Table 2 reports the results of the RGCN on the link prediction task. We observe the highest accuracy on the *hasSpouse* relation, and a drop in performance across the other predicates. We observe the lowest accuracy on the *hasChild* relation.

Additionally, the top half of Table 2 reports the results of GNNExplainer on the task of explainable link prediction. We can see GNNExplainer performed the best on the *hasBrother* predicate explanation in terms of the generalized F_1 score. Note that the RGCN link prediction also performed well on the *hasBrother* predicate. We observe performance drops on the relations *hasChild* and *hasParent*, and on the full dataset, with all predicates included. Indeed the *hasChild* and *hasParent* explanations follow a similar structure and definition of being logically inverse relations of each other.

The top half of Table 2 reports the results of ExplaiNE on the task of explainable link prediction. This method performed the best on the *hasBrother* and *hasSister* predicate subsets in terms of generalized F_1 score. We see the lowest performance on the full datset, followed by the *hasGrandparent* and *hasChild* predicate subsets. Across all metrics and predicate subsets, we find ExplaiNE outperformed GNNExplainer.

Full Data Results. The bottom half of Table 2 further breaks down the results on the full dataset (Full data column of the top half table). We filter the results on the full data for each predicate and compare performance metrics to each predicate subset. For example, the Spouse column from the bottom half of Table 2 reports the benchmark performance of all input triples with the *hasSpouse* predicate from an RGCN trained on the full data. This RGCN is exposed to all possible predicates, whereas the Spouse column from the top half reports benchmark performance on an RGCN trained only on the input triples with the *hasSpouse* predicate.

Comparing the two halves of Table 2, we can see the generalized precision, recall and F_1 scores generally decreased. These

Models	Metrics	Statistics in separated subsets focused on one predicate						
		Spouse	Brother	Sister	Grandparent	Child	Parent	Full data
RGCN	Accuracy	0.903	0.877	0.825	0.787	0.767	0.805	0.81
GNN Explainer	Generalized Precision	0.261	0.366	0.281	0.17	0.137	0.123	0.11
	Generalized Recall	0.434	0.395	0.31	0.17	0.158	0.152	0.121
	Generalized F_1	0.318	0.376	0.291	0.17	0.144	0.133	0.114
	Max-Jaccard	0.275	0.372	0.373	0.137	0.166	0.161	0.11
ExplaiNE	Generalized Precision	**0.296**	**0.407**	**0.353**	**0.21**	**0.181**	**0.202**	**0.173**
	Generalized Recall	**0.546**	**0.458**	**0.459**	**0.21**	**0.223**	**0.243**	**0.2**
	Generalized F_1	**0.378**	**0.424**	**0.388**	**0.21**	**0.195**	**0.216**	**0.182**
	Max-Jaccard	**0.315**	**0.447**	**0.417**	**0.179**	**0.22**	**0.252**	**0.174**

Models	Metrics	Individual predicate statistics on the full dataset						
		Spouse	Brother	Sister	Grandparent	Child	Parent	Full data
RGCN	Accuracy	0.786	0.878	0.826	0.822	0.804	0.8	0.81
GNN Explainer	Generalized Precision	0.071	0.174	0.117	0.129	0.109	0.091	0.11
	Generalized Recall	0.106	0.192	0.142	0.129	0.125	0.102	0.121
	Generalized F_1	0.083	0.18	0.126	0.129	0.114	0.095	0.114
	Max-Jaccard	0.066	0.2	0.151	0.102	0.125	0.12	0.11
ExplaiNE	Generalized Precision	**0.138**	**0.25**	**0.194**	**0.177**	**0.166**	**0.182**	**0.173**
	Generalized Recall	**0.221**	**0.263**	**0.214**	**0.177**	**0.207**	**0.222**	**0.2**
	Generalized F_1	**0.165**	**0.253**	**0.2**	**0.177**	**0.18**	**0.195**	**0.182**
	Max-Jaccard	**0.133**	**0.27**	**0.237**	**0.145**	**0.187**	**0.225**	**0.174**

Table 2: Benchmark results on FrenchRoyalty-200k: Link prediction results for RGCN, and explanation evaluation for GNNExplainer and ExplaiNE. Highest scores per predicate denoted in bold.

Models	Rule	User Scores							
		0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
GNN Explainer	Spouse	0	0	0	50	0	0	**276**	59
	Brother	0	21	0	0	0	2	10	**23**
	Sister	**19**	0	0	0	0	3	13	7
	Grandparent	0	0	0	0	61	504	0	**1104**
	Child	0	0	353	0	0	**585**	0	91
	Parent	0	192	0	0	0	**515**	0	152
	Full data	11	195	312	59	47	**1668**	301	1499
ExplaiNE	Spouse	0	0	0	17	0	0	**327**	47
	Brother	0	**20**	0	0	0	5	7	19
	Sister	**13**	0	0	0	0	6	**13**	10
	Grandparent	0	0	0	0	32	**850**	0	788
	Child	0	0	389	0	0	**516**	0	118
	Parent	0	264	0	0	0	**437**	0	154
	Full data	16	274	336	62	30	**1765**	267	1333

Table 3: Distributions of user scores amongst incomplete attempts. For example, of ExplainE's incorrect predictions on the *hasSpouse* predicate, ExplaiNE unsuccessfully attempted to predict an explanation with a user score of 0.8 **on** 327 **observations.**

large changes across explanation performance metrics suggest that embeddings learned by the RGCN play a significant role. The RGCN trained on the *hasSpouse* subset is learning embeddings using only triples with *hasSpouse* and explanations containing *hasSpouse*, *hasChild*, and *hasParent*. In other words, the RGCN trained on this subset only has access to these predicates. The RGCN trained on the full dataset however has access to all predicates listed in Table 1. This could suggest, for instance, the embeddings from the full data is incorporating additional, useless information into the embedding causing a drop in explanation metrics.

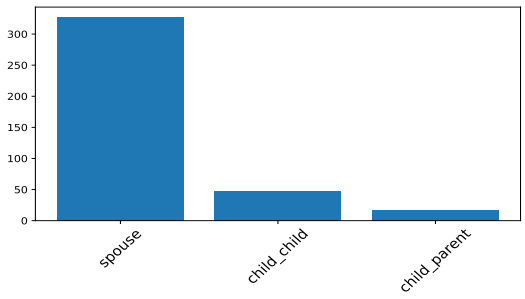

Figure 2: ExplaiNE-Spouse: Most frequently predicted predicates amongst incomplete attempts.

Error Analysis. We define an incomplete attempt to be a predicted explanation where the max-Jaccard score across all possible explanations is less than 1. If two explanations have the same max-Jaccard score, we take the explanation with the highest user score. An incomplete attempt is considered to be a mistake made by the explanation method. Table 3 reports the distributions of user scores amongst the incomplete attempts of GNNExplainer and ExplaiNE for each predicate subset. On the *hasSpouse* subset, GNNExplainer unsuccessfully attempted to predict an explanation with a user score of 0.5 on 50 observations in the test set. From this table, we can see both explanations methods attempted many times but failed to predict explanations with user scores of 0.7. Both explanation methods do not always attempt to predict explanations with the highest user scores (0.9). We recognize the imbalance of user scores, with 0.7 being the most common user score assigned to an explanation. Still, we bring to attention the fact that these explanation methods do not always try to predict the best possible explanation (those with the highest user scores).

Finally, the proposed method and dataset allows us to perform an error analysis on the most frequently predicted predicates amongst incomplete attempts. For instance, Figure 2 shows a histogram of ExplaiNE's incomplete attempts on the *hasSpouse* predicate. The most frequently predicated predicate was *hasSpouse*, accounting for 83% of incomplete attempts. As an example, for an input triple (Eadhild, *hasSpouse*, Hugh the Great), and its ground truth explanations (Hugh the Great, *hasSpouse*, Eadhild), ExplaiNE predicted a first degree neighbor (Hugh the Great, *hasSpouse*, Hedwig of Saxony). This incorrect predicted explanation uses the *hasSpouse* predicate but in the wrong way. This type of analysis can be performed on any predicate. We omit these results due to space constraints.

We can see the importance of the FrenchRoyalty-200k dataset from this benchmark, along with the method we use to construct it, and the metrics we provide. State-of-the-art explanation methods do not always give accurate explanations. Explanation methods must be evaluated with ground truth explanations and quantitative metrics that consider all possible explanations. Our method, dataset, and metrics allow researchers to do so, and to develop new explanation methods and quantitatively evaluate their explanations in a way they were previously unable to.

5 CONCLUSION

On the task of explainable link prediction, there is no standard dataset available with non-unique explanations for quantitative comparisons, as no standard method exists to generate datasets with all possible explanations. Additionally, there is no standard evaluation metric to compare a predicted explanation with all possible ground truths. In this work, we propose a method, including a dataset, FrenchRoyalty-200k, to compare predicted and ground truth explanations when there are multiple ground truths. Furthermore, we propose the use of several evaluation metrics, leveraging the use of graded precision and recall for quantitative comparisons across explanation methods. Lastly, we benchmark two state-of-the-art explanation methods, ExplaiNE and GNNExplainer using the proposed dataset and scoring metrics. Our method can be used to generate other Knowledge Graphs with a variety of different domains, size, density, etc., to support the qualitative and quantitative evaluation of explanations for Relational Graph Convolutional Network link prediction.

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*.
[2] Olivier Corby, Alban Gaignard, Catherine Faron Zucker, and Johan Montagnat. 2012. KGRAM Versatile Inference and Query Engine for the Web of Linked Data. In *IEEE/WIC/ACM Int. Conference on Web Intelligence*.
[3] Jeya Vikranth Jeyakumar, Joseph Noor, Yu-Hsi Cheng, Luis Garcia, and Mani B. Srivastava. 2020. How Can I Explain This to You? An Empirical Study of Deep Neural Network Explanation Methods. In *Advances in Neural Information Processing Systems*.
[4] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2020. A Survey on Knowledge Graphs: Representation, Acquisition and Applications. *CoRR* abs/2002.00388 (2020). arXiv:2002.00388
[5] Bo Kang, Jefrey Lijffijt, and Tijl De Bie. 2019. ExplaiNE: An Approach for Explaining Network Embedding-based Link Predictions. *CoRR* abs/1904.12694 (2019).
[6] Jaana Kekäläinen and Kalervo Järvelin. 2002. Using graded relevance assessments in IR evaluation. *J. Assoc. Inf. Sci. Technol.* (2002).
[7] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Int. Conf. on Learning Representations, ICLR*.
[8] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* (2015).
[9] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *European Semantic Web Conference, ESWC*.
[10] Nava Tintarev and Judith Masthoff. 2012. Evaluating the effectiveness of explanations for recommender systems. *User Modeling and User-Adapted Interaction* (2012).
[11] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *3rd International Conference on Learning Representations, ICLR*.
[12] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. In *Advances in Neural Information Processing Systems*.