Impact of Injecting Ground Truth Explanations on Relational Graph Convolutional Networks and their Explanation Methods for 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 applied to Knowledge Graphs (KGs) for black box link prediction. Several algorithms, or explanations methods, have been proposed to explain the predictions of this model. Recently, researchers have constructed datasets with ground truth explanations for quantitative and qualitative evaluation of predicted explanations. Benchmark results showed state-of-the-art explanation methods had difficulties predicting explanations. In this work, we leverage prior knowledge to further constrain the loss function of RGCNs, by penalizing node embeddings far away from the node embeddings in their associated ground truth explanation. Empirical results show improved explanation prediction performance of state-of-the-art post hoc explanations methods for RGCNs, at the cost of predictive performance. Additionally, we quantify the different types of errors made both in terms of data and semantics.**

***Index Terms*—link prediction, Explainable AI, knowledge graphs, graph neural networks, explanation evaluation**

I. Introduction

Knowledge Graphs [1] often represent facts as triples in the form *(subject, predicate, object)*, where a *subject* and *object* represent an entity, linked by some *predicate*. Knowledge Graphs are often incomplete. Link prediction is performed on Knowledge Graphs to infer new facts from existing ones. Several researchers have proposed to perform link prediction on Knowledge Graphs using graph embedding algorithms. These algorithms 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) should exist between two nodes (entities). A Relational Graph Convolutional Network (RGCN) [2] generalizes Graph Convolutional Networks [3] to Knowledge Graphs, using the scoring function from DistMult [4] as an output layer to return a probability of the input triple being a fact.

RGCNs are treated as a black box, that is, the decision function 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 [6] explains the predictions of any Graph Neural Network, learning a mask over the adjacency matrix to identify the most informative subgraph. ExplaiNE and GNNExplainer return explanations to the user in the form of existing triples in the Knowledge Graph.

Recently, researchers have proposed several datasets with ground truth explanations for link prediction on Knowledge Graphs, allowing for quantitative comparisons of predicted explanations. The Royalty-20k and Royalty-30k datasets [7] were constructed such that each observation has one and only one unique explanation. Researchers may want to know how their explanation method performs when there are multiple correct ways to explain why a relation exists between two nodes. The FrenchRoyalty-200k dataset [8] was constructed to include all possible explanations for each triple in the training and test set. This dataset includes a relevance score between 0 and 1 for each explanation based on how intuitive users found it. For the aforementioned state-of-the-art explanation methods, initial benchmark results showed these methods do not always correctly predict ground truth explanations. Previous approaches to learning Knowledge Graph embeddings did not have access to ground truth explanations, hence do not incorporate information from explanations into the embedding.

In this work, we adapt RGCNs to incorporate prior knowledge from ground truth explanations into each embedding. This is done by constraining the cross entropy loss functions used by RGCNs. We compare several different explanation-constrained loss functions to an RGCN using the standard binary cross entropy. Results show improved predicted explanation performance of post hoc explanation methods for RGCNs, at the cost of predictive performance. Additionally, we quantify the different types of errors made in terms of data and semantics.

II. Related-Work

A. Link Prediction

Knowledge graph embedding algorithms [9] learn continuous vectors for each subject, predicate and object. The loss functions are often designed to capture specific algebraic properties of predicates (symmetric, reflexive, transitive, etc). A scoring function is defined to assign a value to each triple based on if the subject, predicate, and object form a valid fact. Typically, the scoring function is included in the loss function. This paper focuses on performing link prediction using RGCNs.

A Relational Graph Convolutional Network (RGCN) can also learn embeddings and perform link prediction on Knowledge Graphs. The RGCN performs embedding updates for a given entity by multiplying the neighboring entities by a weight matrix for each relation in the dataset, and summing across each neighbor and relation. A weight matrix for self connections is also learned, and added to the neighbor embedding summation. The cross entropy loss function optimized is given by

$$\mathcal{L}_{RGCN} = -\frac{1}{(1+\omega)|\hat{\mathcal{E}}|} \sum_{(s,p,o,y)\in\mathcal{T}} ylog\big(f(s,p,o)\big) + (1-y)log\big(1-f(s,p,o)\big), \tag{1}$$

where $\mathcal{T}$ is the set of all real (positive) and corrupted (negative) triples, ω is the number of negative triples, $|\hat{\mathcal{E}}|$ is the number of unique predicates, f is the function learned by the RGCN, and for positive triples, the label $y = 1$ for positive triples and $y = 0$ for negative triples. This is not the only approach 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 Relational Graph Convolutional Networks.

B. Explainable Link Prediction

Several algorithms have been proposed to explain the predictions of RGCNs. For a 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 the score changes if a link is added or removed between two given nodes.

GNNExplainer [6] explains the predictions of any Graph Neural Network, learning a mask over the input adjacency matrix to identify the most relevant subgraph. GNNExplainer minimizes the cross entropy between the predicted label using the input adjacency matrix, and the predicted label using the masked adjacency matrix.

C. Explanation Aware Loss Function

In the context of Image classification, a recent research paper [10] shows that interpretations are useful and that we can penalize explanations to align neural networks with prior knowledge. To do so, they constrain the loss functions of deep neural networks by introducing an explanation penalty term, which teaches the model to generate correct explanations. This additional constraint was shown to increase classification performance. The explanations generated by this approach however were not empirically evaluated. Without ground truth explanations, this paper relies on assumptions made by either manually annotating explanation labels, or rules to define correct explanations for image data. Indeed manual annotation is difficult with large datasets.

For link prediction on Knowledge Graphs, the standard RGCN optimizes a cross entropy loss function (Equation 1) to learn embeddings. Recently, researchers have used a standard RGCN in a benchmark of three datasets to determine the quality of explanations generated post hoc by GNNExplainer and ExplaiNE [7], [8].

Until recently, benchmarks did not include ground truth for explanations, and the loss functions used by the standard RGCN do not include any constraints that account for them. This lack of constraints on the standard RGCN loss function causes subject and object embeddings in each triple to be mapped far away in the embedding space from the subject and object embeddings in its associated explanation. The Royalty datasets [7], [8] gives us the opportunity to train the predictors with the prior knowledge of ground truth explanations.

D. Our contribution: Explanation-constrained Loss Function for Link Prediction

On the task of link prediction on Knowledge Graphs, the Royalty datasets [7], [8] provided a rule-based approach to generate a ground truth explanation for each input observation. Quantitative and qualitative results showed that both explanation methods did not always generate correct explanations, and performance across multiple metrics were low. In this work, we propose and evaluate several explanation-constrained loss functions to include prior explanation knowledge on the task of RGCN link prediction on Knowledge Graphs. For each triple in the training set, we penalize node embeddings far away from the node embeddings in the associated ground truth explanation. We reproduce the results of [7], [8] on three datasets, Royalty-20k, Royalty-30k, and FrenchRoyalty-200k, and use these to evaluate the impact of different explanation constraint strategies. We find our proposed approaches improve performance of post hoc explanation methods compared to a standard RGCN. Lastly, we perform an error analysis on the Royalty datasets, quantifying errors in terms of both data and semantics. Code for this paper is available online.[1]

III. Injecting Ground Truth Explanations into RGCN Embeddings

A. Constraining the Loss Function

We propose a loss function for RGCNs to improve post hoc explanation method performance on the Royalty datasets. This is achieved by adding an explanation constraint that pushes subject and object embeddings from each input triple closer to

[1] https://github.com/halliwelln/penalized-rgcn

subject and object embeddings in the input triple's explanation. This is captured by the penalty expressed in Equation 2 where, for some input triple $t_p = (s, p, o)$ and an explanation triple $t_j = (s_j, p_j, o_j)$, we propose an explanation aware constraint that can be added to the binary cross entropy used by RGCNs:

$$\begin{aligned}\mathcal{P}(t_p, t_j) = max(||Emb(s) - Emb(s_j)||_2, \\ ||Emb(s) - Emb(o_j)||_2) \\ +max(||Emb(o) - Emb(s_j)||_2, \\ ||Emb(o) - Emb(o_j)||_2).\end{aligned} \tag{2}$$

This penalty sums the maximum ℓ_2 distances between embedding $Emb(.)$ of the subjects and objects of the input triple t_p and an explanation triple t_j. Penalizing the maximum allows us to push the subject embedding $Emb(s)$ from the input triple closer to subject and object embeddings from its ground truth explanation.

The ℓ_2 maximum distance computations accounts for the fact that the direction of the links is an arbitrary modelling decision that should not impact the comparison of the embeddings of subjects and objects. Consider the case when the input triple $t_p = (John, hasParent, Tom)$, and its associated ground truth explanation $t_j = (Tom, hasChild, John)$. Simply summing the distance between subject and objects gives $||Emb(John) - Emb(Tom)||_2 + ||Emb(Tom) - Emb(John)||_2 = 2 * ||Emb(John) - Emb(Tom)||_2$. If however, the direction of the predicate in the explanation changes, for example, if $t_j = (John, isChildof, Tom)$, the distance summation then becomes $||Emb(John) - Emb(John)||_2 + ||Emb(Tom) - Emb(Tom)||_2 = 0$. Certainly the triple pair $(John, hasParent, Tom)$, and $(Tom, hasChild, John)$ contains the same amount of information as $(John, hasParent, Tom)$, and $(John, isChildof, Tom)$, however, a simple summation over distances results in two times the distance or zero. In order to account for this ambiguous case, we compute the maximum between the subject distances and add this to the maximum between the object distances.

We can now augment the standard RGCN binary cross entropy loss with the penalty term in Equation 2 in several ways and to account for several types of prior knowledge from explanation ground truth.

B. Loss Function for Unique Explanations

The Royalty-20k and Royalty-30k datasets [7] contain one and only one unique explanation per predicted triple, describe as the case of non-ambiguous explanations. We introduce the first loss function incorporating the penalty term from Equation 2 for non-ambiguous explanations. Formally, let $t_p \in \mathcal{T}^+$ be a positive triple in the form (s, p, o), let e_p be its explanation that contains a set of explanatory triples t_j for the prediction of t_p. The equation our proposed approach optimizes is given by

$$\mathcal{L}_{sum} = \mathcal{L}_{RGCN} + \frac{1}{|\mathcal{T}^+|} \sum_{t_p \in \mathcal{T}^+} \frac{1}{|e_p|} \sum_{t_j \in e_p} \mathcal{P}(t_p, t_j), \tag{3}$$

where $|.|$ denotes the cardinality, for example $|e_p|$ denotes the number of triples in e_p. Intuitively, Equation 3 takes a training set triple $t_p = (s, p, o)$, and its associated explanation e_p, and applies an ℓ_2 penalty for subject and object embeddings in the explanation of t_p that are far away from t_p's subject and object in the embedding space. In other words, the subject and object embeddings found in each triple's ground truth explanation should be "similar" in the embedding space to the subject and object embeddings, as they are used to explain why a predicate exists between the triple's subject and object. Using the standard RGCN loss function, this relationship between a triple and its ground truth explanation may not be captures in the embedding space without the additional constraint from Equation 3. We apply this loss function to the Royalty-20k and Royalty-30k datasets, results are reported in the following section.

C. Loss Summing all Possible Explanations

The FrenchRoyalty-200k dataset [8] contains multiple explanations for each predicted triples, described as the case of non-unique, or ambiguous explanations. We introduce a loss function including a penalty term for these ambiguous explanations. Formally, let $E_p = \{e_1, ...e_l\}$ be the set of l explanations available for t_p and $e_i = \{(s_1, p_1, o_1), ..., (s_k, p_k, o_k)\}$ be i^{th} explanation for triple t_p. Explanation e_i contains a set of explanatory triples t_j for the prediction of t_p. The proposed loss function is given by

$$\begin{aligned}\mathcal{L}_{sum'} = \mathcal{L}_{RGCN} + \frac{1}{|\mathcal{T}^+|} \sum_{t_p \in \mathcal{T}^+} \\ \frac{1}{|E_p|} \sum_{e_i \in E_p} \frac{1}{|e_i|} \sum_{t_j \in e_i} \mathcal{P}(t_p, t_j)\end{aligned} \tag{4}$$

Similarly, Equation 4 takes a training set triple $t_p = (s, p, o)$, and its associated explanations E_p, and applies an ℓ_2 penalty for subject and object embeddings from all explanations of t_p that are far away from t_p's subject and object in the embedding space. The subtle difference between this loss function and Equation 3 is that $E_p = \{e_p\}$, that is, there is only one explanation available for Equation 3, hence the inner summation can be dropped. Equation 4 however must sum across all explanations available to t_p, hence is used for the FrenchRoyalty-200k dataset.

D. Loss Weighting each Possible Explanations

We also consider a loss function that weights the distance penalty term by the relevance score of each explanation. This approach again pushes the subject and object embeddings of t_p closer to the subject and object embeddings from all triples in E_p. However, this distance penalty term is weighted by

the user score of each explanation in E_p, therefore making the embeddings in t_p more similar to the embeddings from explanations with high relevance scores. This equation is given by

$$\mathcal{L}_{weight} = \frac{1}{|\mathcal{T}^+|} \sum_{t_p \in \mathcal{T}^+} \frac{1}{|E_p|} \sum_{e_i \in E_p} \frac{1}{|e_i|} \sum_{t_j \in e_i} score(e_i) * \mathcal{P}(t_p, t_j) \tag{5}$$

$$\mathcal{L}_{weight} = \mathcal{L}_{weight} + \mathcal{L}_{RGCN} \tag{6}$$

where $score(e_i)$ is the relevance score of e_i taking values between 0 and 1 of the explanation as provided by the FrenchRoyalty-200k dataset. Intuitively, large distances from embeddings in highly relevant explanations are given a larger penalty than large distance from embeddings in less relevant explanations. This loss function considers all triples in the ground truth explanation set E_p, but focuses on intuitive explanations. This loss function relies on the user assigned scores for each explanation included in the FrenchRoyalty-200k, and is thus limited only to applications on this dataset.

E. Loss Selecting the Highest Score

Lastly, we consider a loss function that penalizes subject and object embeddings in t_p that are far away from the subject and object embeddings of the best available explanation e_i, as determined by the given user relevance score. This equation is given by

$$\mathcal{L}_{max} = \frac{1}{|\mathcal{T}^+|} \sum_{t_p \in \mathcal{T}^+} \sum_{t_j \in e_i; score(e_i) = \max\limits_{e \in E_p} score(e)} \frac{\mathcal{P}(t_p, t_j)}{|e_i|} \tag{7}$$

$$\mathcal{L}_{max} = \mathcal{L}_{max} + \mathcal{L}_{RGCN} \tag{8}$$

This loss function pushes the subject and object embeddings from t_p close to the subject and object embeddings in the best explanation e_i in the embedding space, making these embeddings more similar to each other than the standard RGCN. Embeddings from all other available ground truth explanations are not factored in. Similar to Equation 6, this loss function relies on the user assigned scores from the FrenchRoyalty-200k, hence this loss function is limited only to applications on this dataset.

IV. Results and Evaluations

In this section, we evaluate the proposed loss functions on three datasets. The Royalty-20k dataset contains 3 types of predicates: *hasSpouse*, *hasSuccessor*, and *hasPredecessor*. The Royalty-30k dataset also contains 3 types of predicates including *hasSpouse*, *hasGrandparent*, and *hasParent*, where *hasParent* is only used to explain *hasGrandparent*. These datasets are used to evaluate explanation quality when there is one and only one explanation for each predicted triple. The FrenchRoyalty-200k contains 6 types of predicates also based on family relations, *hasSpouse hasBrother*, *hasSister*, *hasGrandparent*, *hasChild*, and *hasParent*. Each predicted triple in this dataset includes all possible explanations, and is used to evaluate explanation quality when there are multiple to choose from. We compare all loss functions with a standard RGCN (using the loss function from Equation 1). We apply two state-of-the-art explanation methods, GNNExplainer [6] and ExplaiNE [5] to all RGCNs post hoc, and compare the quality of explanation generated by GNNExplainer and ExplaiNE.

For all experiments in this work, we fix the number of embedding dimensions to 10 as done in the original benchmark. Across all three datasets, we subset the data by each predicate, and report results on each subset. For example, on the Royalty-20k dataset, the Spouse column from Table I gives performance results on a subset of data using only *hasSpouse* triples and their associated explanations. Additionally, we report results on the full dataset, with all predicates included.

On the Royalty-20k and Royalty-30k datasets, predicted explanation performance is measured using the Jaccard score between each predicted and ground truth explanation. We also report precision, recall and F_1 scores, however, the original authors recommend measuring explanation quality on these datasets using the Jaccard score. On the FrenchRoyalty-200k dataset, predicted explanation performance is measured using the Generalized Precision, Recall, and F_1 scores [11], along with the Max-Jaccard score [8].

A. Results with Non-Ambiguous Explanations

The top two rows of Table I report the link prediction results for the standard RGCN and the loss function in Equation 3 on the Royalty-20k dataset. We can see the standard RGCN outperformed the proposed approach on the full dataset, along with the *hasSpouse* subset. The proposed approach outperformed the standard RGCN on the *hasSuccessor* and *hasPredecessor* subsets.

Rows three and four of Table I report the results of GNNExplainer applied to a standard RGCN, and applied to the proposed RGCN in Equation 3 on the task of explainable link prediction. We observe the GNNExplainer applied to the proposed RGCN outperformed or matched the GNNExplainer applied to the baseline in terms of the Jaccard score on all subsets, and the full dataset.

Rows five and six of Table I report the results of ExplaiNE applied to a standard RGCN, and applied to the proposed RGCN in Equation 3. On the *hasSpouse*, *hasSuccessor* and *hasPredecessor* subsets, we find ExplaiNE when applied to the RGCN in Equation 3 improved all performance metrics. On the full dataset, we found using the proposed approach resulted in an improved Jaccard score.

The three rightmost columns of Table I report the performance metrics for the standard RGCN and the proposed approach on the Royalty-30k dataset. On the task of link prediction, we again find the proposed approach decreased the accuracy on all subsets, including the full dataset.

Models	Metrics	Royalty-20k Results				Royalty-30k Results		
		Spouse	Successor	Predecessor	Full data	Spouse	Grandparent	Full data
RGCN	Accuracy	**0.737**	0.612	0.683	**0.77**	**0.737**	**0.654**	**0.687**
$\mathcal{L}_{sum}$	Accuracy	0.517	**0.989**	**0.717**	0.758	0.517	0.643	0.678
GNN Explainer								
with RGCN	Precision	**0.657**	0.154	0.123	0.273	**0.657**	0.064	0.258
	Recall	**0.498**	0.151	0.121	0.22	**0.498**	0.089	0.297
	F_1	**0.567**	0.153	0.122	0.251	**0.567**	0.074	0.276
	Jaccard	**0.34**	0.149	0.119	0.193	**0.34**	0.089	0.114
with $\mathcal{L}_{sum}$	Precision	**0.657**	**0.18**	**0.133**	**0.275**	**0.657**	**0.066**	**0.259**
	Recall	**0.498**	**0.176**	**0.13**	**0.234**	**0.498**	**0.092**	**0.298**
	F_1	**0.567**	**0.178**	**0.131**	**0.253**	**0.567**	**0.077**	**0.277**
	Jaccard	**0.34**	**0.171**	**0.128**	**0.194**	**0.34**	**0.092**	**0.154**
ExplaiNE								
with RGCN	Precision	0.886	0.28	0.178	0.574	0.886	0.104	0.427
	Recall	0.668	0.272	0.176	0.493	0.668	0.139	0.471
	F_1	0.762	0.276	0.177	0.531	0.762	0.119	0.448
	Jaccard	0.45	0.264	0.174	0.412	0.45	0.139	0.185
with $\mathcal{L}_{sum}$	Precision	**0.949**	**0.402**	**0.257**	**0.582**	**0.949**	**0.123**	**0.453**
	Recall	**0.712**	**0.391**	**0.251**	**0.501**	**0.712**	**0.164**	**0.506**
	F_1	**0.813**	**0.396**	**0.254**	**0.539**	**0.813**	**0.141**	**0.478**
	Jaccard	**0.474**	**0.38**	**0.245**	**0.419**	**0.474**	**0.164**	**0.286**

TABLE I: Results on Royalty-20k, Royalty-30k datasets: Link prediction results for baseline RGCN and proposed loss functions, along with explanation evaluation for GNNExplainer and ExplaiNE. Highest scores per predicate denoted in bold.

Rows three and four of Table I report the results of GNNExplainer applied to the baseline RGCN, and proposed RGCN on the task of explainable link prediction. We find equal or better performance across all metrics. The precision, recall, and F_1 score remain relatively unchanged on the full dataset.

Rows five and six of Table I report the results of ExplaiNE applied to the baseline RGCN, and the proposed RGCN. Here we see improved performance on all metrics, and across all data subsets, including the full dataset.

B. Results with Non-Unique Explanations

The top four rows of Table II report the link prediction results of the baseline and proposed methods. In general, the baseline RGCN from Equation 1 outperformed the proposed methods in terms of accuracy on the task of link prediction, with the exception of the *hasSpouse*, *hasSister*, and *hasChild* subsets.

Rows five through eight of Table II report the results of GNNExplainer applied the baseline RGCN, and the proposed approaches from Equations 4, 6, and 8 on the task of explainable link prediction. Overall, we found all approaches had similar performance metrics, with two proposed approaches having a small increase in Max-Jaccard score on the full dataset.

Rows nine through twelve of Table II report the results of ExplaiNE applied to the baseline RGCN, and the proposed approaches on the task of explainable link prediction. We found the proposed approach improved performance on almost all metrics. Most notably, a large increase in Max-Jaccard score on the full dataset and *hasSister* subset.

V. Error Analysis: Quantitative Evaluation of Explanations

A. Royalty-20k

The top row of Table III gives a breakdown of each explanation method's most frequent error by subset for the Royalty-20k dataset when applied to $\mathcal{L}_{RGCN}$ and $\mathcal{L}_{sum}$. Each row reports the most frequent predicate, and the percentage of errors this predicate occured in. For example, under the *hasSpouse* subset, the most common predicate across ExplaiNE's incorrectly predicted explanations (when applied to the RGCN in Equation 3) was *hasSpouse*, and this predicate was observed in 100% of errors. This error occurs when ExplaiNE predicts the wrong subject or object in the explanation. This can occur on the *hasSpouse* subset, as under this subset, there is only one possible predicate to predict (*hasSpouse*).

On the Royalty-20k dataset, we can see on the *hasSuccessor* subset that the 94% of ExplaiNE with $\mathcal{L}_{sum}$ errors contained the *hasPredecessor* predicate. This type of error occurs when the subject and/or object in the predicted explanation are incorrect. We can deduce this due to the fact that on the *hasSuccessor* dataset, the RGCNs and explanation methods only observe two predicates, *hasSuccessor* and *hasPredecessor*. GNNExplainer when applied to both RGCNs however produce more uniform errors, where 52% of errors occurred by using the wrong subject and/or object, and the remaining errors occurred by identifying the wrong predicate. For GNNExplainer applied to both RGCNs, we observe a similar phenomenon on the *hasPredecessor* subset as well.

Note there are three types of explanation errors, one where the predicate in the predicted explanation is incorrect, one where the subject and/or object in the predicted explanation is incorrect, or both. From Tabe III, we can see that ExplaiNE, when applied to the RGCN from $\mathcal{L}_{sum}$, has an increased number of errors using the wrong subject and object on the *hasSuccessor* subset. Recall each *hasSuccessor* predicate has an associated *hasPredecessor* ground truth. Here, the proposed approach produces more errors using the *hasPredecessor* predicate.

The first row of Table IV reports the most frequently missing predicate from the explanation method's errors for the Royalty-

Models	Metrics	FrenchRoyalty-200k Results						
		Spouse	Brother	Sister	Grandparent	Child	Parent	Full data
$\mathcal{L}_{RGCN}$	Accuracy	0.935	**0.909**	0.853	**0.858**	0.792	**0.838**	**0.928**
$\mathcal{L}_{sum'}$	Accuracy	**0.973**	0.864	**0.999**	0.599	0.8	0.639	0.877
$\mathcal{L}_{max}$	Accuracy	0.966	0.75	0.824	0.648	0.793	0.697	0.878
$\mathcal{L}_{weight}$	Accuracy	0.966	0.909	0.971	0.615	**0.801**	0.706	0.897
GNN Explainer								
with $\mathcal{L}_{RGCN}$	G. Precision	**0.246**	0.323	**0.34**	0.162	0.142	**0.131**	0.109
	G. Recall	**0.415**	0.333	0.353	0.162	0.167	**0.154**	**0.119**
	G. F_1	**0.302**	0.327	0.344	0.162	0.15	**0.139**	0.112
	Max-Jaccard	**0.256**	**0.345**	0.299	0.128	0.16	**0.161**	0.109
with $\mathcal{L}_{sum'}$	G. Precision	0.243	**0.324**	**0.34**	0.162	0.142	0.13	0.108
	G. Recall	0.411	**0.335**	0.353	0.162	0.166	**0.154**	0.118
	G. F_1	0.299	0.328	0.344	0.162	0.14	0.138	0.111
	Max-Jaccard	0.254	**0.345**	0.299	0.128	0.161	0.16	0.109
with $\mathcal{L}_{max}$	G. Precision	**0.246**	**0.324**	**0.34**	**0.164**	0.143	0.128	0.108
	G. Recall	0.414	**0.335**	0.353	**0.164**	0.167	0.151	0.118
	G. F_1	**0.302**	0.328	0.344	**0.164**	0.151	0.136	0.112
	Max-Jaccard	0.255	**0.345**	0.299	**0.13**	0.161	0.159	**0.11**
with $\mathcal{L}_{weight}$	G. Precision	0.243	**0.324**	0.328	0.163	**0.144**	0.13	**0.11**
	G. Recall	0.411	**0.335**	0.342	0.163	**0.168**	0.153	0.12
	G. F_1	0.299	0.328	0.333	0.163	**0.152**	0.138	**0.113**
	Max-Jaccard	0.254	**0.345**	0.299	0.129	**0.162**	**0.161**	**0.11**
ExplaiNE								
with $\mathcal{L}_{RGCN}$	G. Precision	0.336	0.48	0.379	0.234	0.221	0.255	**0.192**
	G. Recall	0.637	0.49	0.418	0.234	0.279	0.27	0.218
	G. F_1	0.435	0.483	0.392	0.234	0.24	0.26	0.2
	Max-Jaccard	0.363	0.504	0.417	0.201	0.241	0.27	0.193
with $\mathcal{L}_{sum'}$	G. Precision	**0.37**	**0.585**	**0.536**	**0.244**	**0.231**	0.247	0.188
	G. Recall	**0.726**	**0.605**	**0.667**	**0.244**	**0.291**	0.258	**0.232**
	G. F_1	**0.488**	**0.591**	**0.58**	**0.244**	**0.251**	0.25	**0.203**
	Max-Jaccard	**0.41**	**0.598**	**0.539**	**0.211**	**0.246**	0.273	**0.209**
with $\mathcal{L}_{max}$	G. Precision	0.338	0.433	0.487	0.224	0.224	**0.267**	0.177
	G. Recall	0.66	0.471	0.592	0.224	0.283	**0.286**	0.213
	G. F_1	0.444	0.444	0.522	0.224	0.243	**0.273**	0.189
	Max-Jaccard	0.377	0.523	0.402	0.189	0.242	**0.303**	0.196
with $\mathcal{L}_{weight}$	G. Precision	0.351	0.538	0.485	0.232	0.227	0.263	0.189
	G. Recall	0.69	0.567	0.629	0.232	0.287	0.275	0.23
	G. F_1	0.463	0.547	0.533	0.232	0.247	0.267	**0.203**
	Max-Jaccard	0.39	0.557	0.407	0.196	0.243	0.295	0.208

TABLE II: Results on FrenchRoyalty-200k: Link prediction results for baseline RGCN and proposed model, along with explanation evaluation for GNNExplainer and ExplaiNE. Highest scores in bold, and G. being an abbreviation for Generalized.

20k dataset. Each row denotes the predicate subset, the ground truth predicates defining the rule, and the percentage of triples not containing the ground truth predicate(s). For example, under the *hasSuccessor* subset of the Royalty-20k dataset, 6% of ExplaiNE's errors (when applied to $\mathcal{L}_{sum}$) did not contain *hasPredecessor*.

B. Royalty-30k

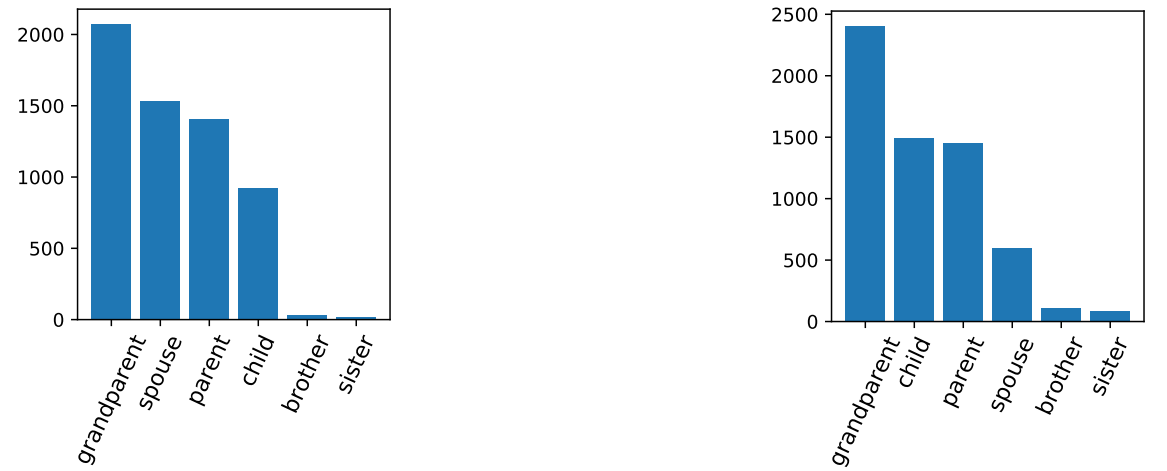
The second row of Table III gives a breakdown of each explanation method's most frequent error by subset for the Royalty-30k dataset when applied to $\mathcal{L}_{RGCN}$ and $\mathcal{L}_{sum}$. After applying ExplaiNE to $\mathcal{L}_{sum}$, we can see on the *hasGrandparent* subset, the most frequently predicted predicate was *hasParent*, accounting for 56% of errors. Conversely, for ExplaiNE with the baseline $\mathcal{L}_{RGCN}$, the most frequently predicted predicate is *hasGrandparent*. We can conclude from this that the explanation aware loss function $\mathcal{L}_{sum}$ changed the most frequent type of error made by ExplaiNE. Rather than predict the wrong predicate, the explanation aware loss instead produces errors using the correct predicate but wrong subject and/or objects.

The second row of Table IV reports the most frequently missing predicate from each explanation method's errors for the Royalty-30k dataset. On the *hasGrandparent* subset, we find a decreased number of errors missing the *hasParent* explanation, consistent with Table III. In general, we found GNNExplainer when applied to an explanation aware RGCN had minimal changes in errors metrics.

C. FrenchRoyalty-200k

The last row of Table III gives a breakdown of each explanation method's most frequent error by subset for the FrenchRoyalty-200k dataset when applied to $\mathcal{L}_{RGCN}$ and $\mathcal{L}_{sum'}$. On the *hasBrother* subset, we can see the errors produced by ExplaiNE with $\mathcal{L}_{sum'}$ results in errors using the *hasParent* predict, instead of *hasGrandparent* produced by the baseline.

Figures 1a and 1b show frequency counts of the most frequently predicted predicates amongst predictions made by $\mathcal{L}_{sum'}$ with a Max-Jaccard score less than 1. We can see both ExplaiNE and GNNExplainer's most frequently predicte predicate is *hasGrandparent*. Additionally, both explanation methods least frequently predicted predicate amongst errors were *hasBrother*, and *hasSister*. We found ExplaiNE had difficulty predicting *hasSpouse* explanations, while GNNExplainer had fewer *hasSpouse* errors, and more errors with *hasChild* explanations. The number of errors made by GNNExplainer on the *hasParent* and *hasChild* subsets were nearly equal. We



(a) FrenchRoyalty-200k ExplaiNE

(b) FrenchRoyalty-200k GNNExplainer

Fig. 1: RGCN with $\mathcal{L}_{sum}$: Predicate Frequency Count on Incorrectly Predicted Explanations on FrenchRoyalty-200k Dataset.

		Most Frequently Predicated Predicate							
		ExplaiNE with $\mathcal{L}_{sum}$		ExplaiNE with $\mathcal{L}_{RGCN}$		GNNExplainer with $\mathcal{L}_{sum}$		GNNExplainer with $\mathcal{L}_{RGCN}$	
Dataset	Predicate	Most Frequent Predicate	% of Error	Most Frequent Predicate	% of Error	Most Frequent Predicate	% of Error	Most Frequent Predicate	% of Error
$Royalty-20k$	*hasSpouse*	*hasSpouse*	100%	*hasSpouse*	100%	*hasSpouse*	100%	*hasSpouse*	100%
	hasSuccessor	*hasPredecessor*	94%	*hasPredecessor*	67%	*hasSuccessor*	52%	*hasPredecessor*	52%
	hasPredecessor	*hasPredecessor*	64%	*hasPredecessor*	55%	*hasPredecessor*	57%	*hasPredecessor*	51%
$Royalty-30k$	*hasSpouse*	*hasSpouse*	100%	*hasSpouse*	100%	*hasSpouse*	100%	*hasSpouse*	100%
	hasGrandparent	*hasParent*	56%	*hasGrandparent*	55%	*hasGrandparent*	64%	*hasGrandparent*	64%
$FrenchRoyalty$ $200k$	*hasSpouse*	*hasSpouse*	92%	*hasSpouse*	84%	*hasSpouse*	51%	*hasSpouse*	50%
	hasBrother	*hasParent*	72%	*hasGrandparent*	53%	*hasGrandparent*	47%	*hasGrandparent*	45%
	hasSister	*hasParent*	60%	*hasParent*	53%	*hasGrandparent*	32%	*hasGrandparent*	32%
	hasGrandparent	*hasGrandparent*	44%	*hasGrandparent*	56%	*hasGrandparent*	57%	*hasGrandparent*	56%
	hasChild	*hasParent*	30%	*hasParent*	33%	*hasGrandparent*	41%	*hasGrandparent*	41%
	hasParent	*hasParent*	45%	*hasParent*	46%	*hasGrandparent*	37%	*hasGrandparent*	38%

TABLE III: Most frequent predicate across incorrectly predicted explanations, along with the percentage of error by subset. Note $\mathcal{L}_{sum'}$ is used for FrenchRoyalty-200k.

omit this analysis on Royalty-20k and Royalty-30k datasets due to space constraints.

VI. Discussion

On all three datasets, we found the proposed approaches matched or increased the Jaccard (or Max-Jaccard) scores on ExplaiNE when training on the full dataset with all predicates included. We found however, the baseline RGCN outperformed the proposed approach on the task of link prediction on the same datasets. From these experiments, we observe a trade off between black box model performance and explainability. Including prior information from ground truth explanations into the embeddings of RGCNs improves the quality of explanations generated by ExplaiNE and GNNExplainer. However, this comes at the cost of predictive power. Our approach allows practitioners and researchers to find a balance between predictive power and model explainability that the standard RGCN is unable to provide.

Additionally, we found our approach had the biggest impact on ExplaiNE's explanations, and a minimal impact on GNNExplainer's explanations. Understanding why the proposed approach had a larger impact on ExplaiNE's performance metrics than that of GNNExplainer would require a further understanding of what properties of the graph the embedding has learned. We leave this task for future work.

We recognize the difficulties in predicting explanations, even after making improvements, Jaccard (and Max-Jaccard) scores were still low. In fact, we found many of the Jaccard scores to be less than 0.5. Applying explanation methods post hoc to a black box model creates difficulties in diagnosing errors in predicted explanations, as there are many possible sources of error. When an explanation method produces an incorrectly predicted explanation, there are no available techniques to our knowledge that can identify if the explanation method is flawed, or if the error is due to a bad embedding that has not capturing the necessary information. Recent research has raised a similar concern, and that explanation methods for black boxes can be misleading [12]–[15].

This work contributes to being able to identify where in the pipeline errors are caused. Injecting knowledge into the graph

Most Frequently Missing Predicate						
Dataset	Predicate	Ground Truth	ExplaiNE with $\mathcal{L}_{sum}$ % Missing	ExplaiNE with $\mathcal{L}_{RGCN}$ % Missing	GNNExplainer with $\mathcal{L}_{sum}$ % Missing	GNNExplainer with $\mathcal{L}_{RGCN}$ % Missing
$Royalty-20k$	*hasSpouse*	*hasSpouse*	0%	0%	0%	0%
	hasSuccessor	*hasPredecessor*	6%	33%	52%	48%
	hasPredecessor	*hasSuccessor*	64%	55%	57%	49%
$Royalty-30k$	*hasSpouse*	*hasSpouse*	0%	0%	0%	0%
	hasGrandparent	*hasParent*	44%	55%	64%	64%
		hasParent	44%	55%	64%	64%

TABLE IV: Most frequently missing predicate. Each row denotes the predicate subset, the ground truth predicates defining the rule, and the percentage of triples not containing the ground truth predicate(s)

embedding shows GNNExplainer's errors are likely due to its parameters learned and not the RGCN embeddings, where as ExplaiNE's error are due to the embeddings. We recognize that the proposed approaches do not often have a practical application due to the difficulty of obtaining ground truth explanations. Rather, this work shows the theoretical impact of using prior knowledge during training, in order to identify if errors come from the RGCN or the post hoc explanation methods. For instance, this work showed that injecting the ground truths cause an accuracy decrease in some cases, and this opens some new research directions.

The lack of significant changes in performance metrics of GNNExplainer is likely due to the large number of parameters used by the model for each observation. Perturbations to the RGCN embedding are less influential on the predicted explanation, hence we can conclude GNNExplainer is less dependent on the RGCN embeddings for explanation predictions than ExplaiNE.

We are aware that there are few Knowledge Graphs providing a ground truth for explanations, however we wanted to evaluate the impact of such knowledge on different methods before investing resources in campaigns to manually annotate Knowledge Graphs with explanations. This work focuses on the case of supervised explanation prediction, where ground truth explanations are available. We provide a theoretical study of the behaviour of several explanation methods in the presence of explanation aware embeddings.

VII. Conclusion

In this work, we apply an explanation-constrained loss function [10] to RGCNs for link prediction on Knowledge Graphs. We add a penalty term for subject and object embeddings far away from the subject and object embeddings found in the ground truth explanation. We compare several different explanation-constrained loss functions to a baseline RGCN, and evaluate performance on three datasets with ground truth explanations. Results show improved performance of post hoc explanation methods. We perform an error analysis on the Royalty datasets, quantifying errors in terms of both data and semantics. This work provides opportunity for future extensions, such as leveraging the proposed RGCNs to distinguish between model error and an explanation method error. That is, determining if an incorrect explanation is caused by the embedding learned by the RGCN, or if the error is caused by the explanation method.

References

[1] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, *et al.*, "Knowledge graphs," *preprint arXiv:2003.02320*, 2020.

[2] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European Semantic Web Conference, ESWC*, 2018.

[3] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations, ICLR*, 2017.

[4] B. Yang, W. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *International Conference on Learning Representations, ICLR*, 2015.

[5] B. Kang, J. Lijffijt, and T. D. Bie, "Explaine: An approach for explaining network embedding-based link predictions," *CoRR*, vol. abs/1904.12694, 2019.

[6] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," in *Advances in Neural Information Processing Systems*, 2019.

[7] N. Halliwell, F. Gandon, and F. Lecue, "Linked Data Ground Truth for Quantitative and Qualitative Evaluation of Explanations for Relational Graph Convolutional Network Link Prediction on Knowledge Graphs," in *International Conference on Web Intelligence and Intelligent Agent Technology*, (Melbourne, Australia), Dec. 2021.

[8] N. Halliwell, F. Gandon, and F. Lecue, "User Scored Evaluation of Non-Unique Explanations for Relational Graph Convolutional Network Link Prediction on Knowledge Graphs," in *International Conference on Knowledge Capture*, (Virtual Event, United States), Dec. 2021.

[9] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition and applications," *CoRR*, vol. abs/2002.00388, 2020.

[10] L. Rieger, C. Singh, W. J. Murdoch, and B. Yu, "Interpretations are useful: Penalizing explanations to align neural networks with prior knowledge," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, 2020.

[11] J. Kekäläinen and K. Järvelin, "Using graded relevance assessments in IR evaluation," *J. Assoc. Inf. Sci. Technol.*, 2002.

[12] C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova, and C. Zhong, "Interpretable machine learning: Fundamental principles and 10 grand challenges," *CoRR*, vol. abs/2103.11251, 2021.

[13] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nat. Mach. Intell.*, vol. 1, no. 5, pp. 206–215, 2019.

[14] T. Laugel, M. Lesot, C. Marsala, X. Renard, and M. Detyniecki, "The dangers of post-hoc interpretability: Unjustified counterfactual explanations," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI*, 2019.

[15] H. Lakkaraju and O. Bastani, ""how do I fool you?": Manipulating user trust via misleading black box explanations," in *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA*, 2020.