The relations that make a difference in a relational world

An implementation and exploration of RGCN Explainer

Teresa Liberatore[1]

Vrije Universiteit Amsterdam, Triply [1]

Abstract. The Relational Graph Convolutional Network (RGCN) model is becoming popular for performing Machine Learning on Knowledge Graphs (KGs). However, there aren't many tools available yet to help understand and explain how the model works.
This thesis introduces two approaches to enhance the interpretability of the RGCN model in the context of node classification tasks. The first method, RGCNExplainer, extends the capabilities of GNNExplainer to handle relational data. The second method, the Relation Attribution method, aims to identify the key relations that significantly influence the model's decision-making process for each target class.
These methods are evaluated on four Knowledge Graph datasets, demonstrating their capacity to offer valuable insights into the RGCN model by leveraging the semantic information embedded within the graph relations. Through these enhancements, the thesis contributes to advancing the explainability and transparency of the RGCN model, enabling a better understanding of its decision mechanisms.

Keywords: RGCN · Explainable AI · GNNExplainer · Relational Data · Knowledge Graphs

1 Introduction

Graphs are ubiquitous and used across many domains to represent structured and relational data, such as Social Networks, Biological Networks or open/closed-domain Knowledge Bases [2]. Among them, Knowledge Graphs (KGs) are graph-structured datasets, representing entities and relations between pairs of entities, allowing for different semantics to be embedded in the relations and types of the entities [3].

The ability to perform statistical relational learning over such data enables inference over link properties and types, whose outcomes find applications in many fields ranging from Natural Language Processing to Computer Vision [4]. Inferences over link properties involve tasks like predicting missing connections between entities. For instance, it could entail predicting that "Actor X" participated in "Movie Y," leveraging the structure of the knowledge graph. Conversely, inferences about type properties involve predicting the category that certain entities belong to. For example, it could include predicting that "Movie Y" falls under the genre "Z," which is commonly called node classification.

In particular, Relational Graph Convolution Networks (RGCNs) have been introduced in [5] : they are message passing frameworks for learning valuable latent features of relational graphs - which have become widely adopted to perform Machine Learning over KGs. In particular, RGCN achieved SOTA results on the downstream task of node classification over knowledge graphs, obtaining, for example, more than 95% accuracy on the benchmark AIFB dataset [6] [7]. The success of the RGCN model can also be measured in terms of the number of citations that the original paper, [5] - 3603 in June 2023 - obtained in the past 5 years.

Despite achieving good results, RGCN models still lack explainability. This is because there exists no dedicated tool to explain the black box of mechanisms underlying the RGCNs .

While various methods have been developed to make Graph Neural Networks (GNNs) explainable [8], there is a noticeable absence of tools specifically designed for explaining GNNs that operate with relational graphs, like RGCNs. Given that Knowledge Graphs (KGs) naturally offer background knowledge in a format machines can understand [9], there is a belief that a tool designed specifically to explain RGCNs could combine KG characteristics with existing approaches used for explaining GNNs. This integration could offer more meaningful, insightful, and reliable explanations.

RGCNExplainer, a model-specific explainability tool for RGCN is thus introduced in this work, where the neuro-symbolic nature of RGCN is leveraged to enhance the understanding of both the inner structure of the input graph and of the model that tries to learn this inner structure, the RGCN. In particular, the proposed method is a direct *semantic* expansion of GNNExplainer, a perturbation-based GNN explainability method that outputs instance-level explanations, by learning graph soft masks via mask optimization [10].

The decision of starting from GNNExplainer was driven by its perturbation-based nature, enabling experimentation with various types of perturbations and the incorporation of prior domain or statistical knowledge.

Relation Attribution Approaches are also introduced to explore how different relationship types influence the classification of nodes into various classes of interest. Specifically, these approaches treat different relations as features and analyze their effects on predictions.

Motivation The objective of this project is driven by the notion that Linked Data represented by relational graphs is likely to become the norm for storing and sharing information in the future. In such a scenario, it is crucial to have models that perform Machine Learning over KGs and are easily comprehensible by humans. Additionally, exploring the semantics learned by these models raises questions about how humans categorize and classify entities compared to the network's approach. This is an intriguing topic in ontology engineering, where grouping entities into concepts and establishing relationships between them is crucial.

Under this perspective this work becomes relevant for Triply [1], the company that fully supports the investigation of this topic. Triply aims to simplify the use of Linked Data in practical and large-scale applications and is thus interested in supporting research in this field. In particular, due to the ever-growing use of Machine Learning to make inferences and predictions, it becomes relevant to research methods that allow for explainable ML on Knowledge Graphs.

Consider a scenario where a client has a dataset containing incomplete details about specific entities. In this situation, the RGCN model can be trained and applied to forecast those missing details. To ensure comprehensible explanations for these predictions, RGCNExplainer comes into play. This becomes particularly crucial if the predicted information holds a degree of sensitivity.

Research Questions By implementing RGCNExplainer and exploring the obtained explanation results on both small benchmark and larger datasets, this work addresses two main research questions, which can be categorized as follows:

1. **Implementation of RGCNExplainer and Exploration of the impact of knowledge injection on RGCN explanations**
   (a) How can we effectively generate explanations for the RGCN model in the node classification task? Specifically, how can we identify the most influential edges contributing to the model's predictions for a given node?
   (b) Does incorporating prior statistical and domain knowledge of the input graph improve the quality and interpretability of RGCN explanations? How does the injection of such knowledge affect the explanation process and the fidelity of the generated explanations?
2. **Implementation of the Relation Attribution method: Investigating the relations that define a class**
   (a) For each class of interest which relations make a difference in determining the class membership of nodes? In other words, which relations provide the most crucial information to the RGCN model when classifying nodes into specific classes? Can understanding the role of these relations provide valuable insights into the model's decision-making process and improve our comprehension of the classification outcomes?

Contribution The main contributions of this work can be summarized as follows:

1. The implementation of RGCNExplainer: an explainability tool designed to give explanations to message-passing models for relational graphs such as RGCN
2. An exploration of the explanations obtained with different degrees of prior knowledge injected into the explanations
3. The implementation of the Relation Attribution method to identify the relations that play a critical role in determining the membership of nodes in specific classes

In summary, this work contributes to the fields of explainability for RGCN models, understanding the impact of prior knowledge on explanations, and investigating the role of relations in node classification tasks.

2 Background and Related Work

This section provides background on Knowledge Graphs (KGs), Relational Graph Convolutional Networks (RGCNs), approaches for explainable GNNs and an overview about Explainable AI with Semantic Web technologies.

2.1 Knowledge Graphs

Knowledge Graphs are generally defined as a data structure describing entities and their relationships by means of direct, edge-labeled graphs, often organizing them in an ontological schema. In particular, the set of concepts and properties between them is referred to as Terminology Box (TBox), whilst the set of statements about individuals belonging to those concepts as Assertion Box (ABox) [9].

Nodes in the graph represent entities, whereas the edges between nodes represent various kinds of relations that exist between entities. A Knowledge Graph can thus be defined as a directed graph with labeled vertices and edges. Formally, a KG can be defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where $\mathcal{R}$ represents the set of edge labels (relations) and $< s, r, o > \in \mathcal{E}$ is a set of tuples representing that a subject entity s and an object entity o are connected by the relation $r \in \mathcal{R}$ [11].

In practical terms, Knowledge Graphs serve as a structured repository for organizing and preserving factual information, facilitating a wide array of applications spanning from question answering to information retrieval. However, even extensive KGs like Wikidata [12] and DBpedia [13] exhibit incompleteness, which negatively impacts downstream applications. Predicting missing information in KGs is thus the main focus of statistical relational learning (SRL) [5].

2.2 Relational Graph Convolutional Networks

Relational Graph Convolutional Networks, RGCNs, are an example of SRL, being an extension of Graph Convolutional Networks, GCNs, that operate on local graph neighborhoods [14] to large-scale relational data.

GCNs can be understood as special cases of the simple differentiable message-passing framework, where the hidden states of neighboring nodes of node v_i form the message (each node representation is replaced by the average of its neighbors) passed to node v_i through an element-wise activation function. This type of transformation has been shown to be effective at accumulating and encoding features from local, structured neighborhoods.

Motivated by these architectures, the propagation model for calculating the forward-pass update of an entity/node in a relational directed and labeled multi-graph is defined as follow [5] :

$$h_i^{(l+1)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{j\in\mathcal{N}_i^r}\frac{1}{c_{i,r}}W_r^{(l)}h_j^{(l)} + W_0^{(l)}h_i^{(l)}\right) \quad (1)$$

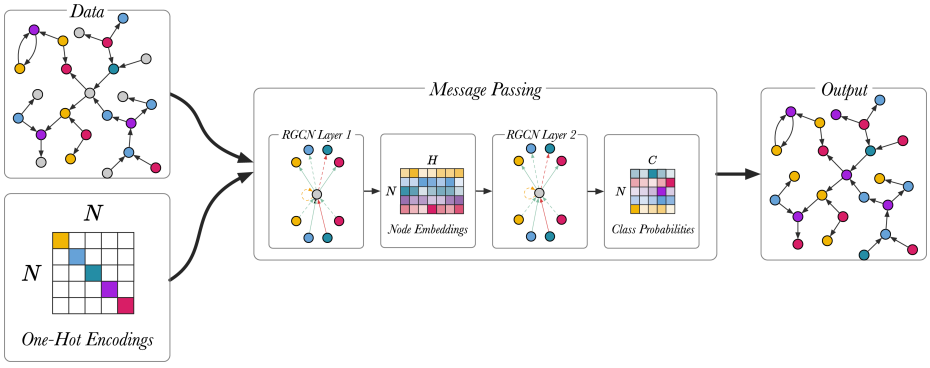
where N_i^r denotes the set of neighbor indices of node i under relation $r \in \mathcal{R}$. Indeed Eq.1 accumulates transformed vectors of neighboring nodes through a normalized sum, where relation-specific transformations, depending on the type and direction of an edge. To ensure that the representation of a node at layer l+1 can also be informed by the corresponding representation at layer l, a single self-connection of a special relation type is added to each node in the data.

An RGCN layer update consists in evaluating Eq.1 in parallel for every node in the graph, where in order to make the matrix multiplication more efficient, a sparse matrix representation is introduced. Multiple layers are stacked to allow for dependencies across several relational steps, where each relational step is usually called a *hop*. In that sense, the overall RGCN model takes the form of L stacked layers as defined in Eq.1 where the output of the previous layer is the input of the next layer.

In particular, for the supervised classification of nodes/entities, RGCN layers in the form of Eq.1 are stacked together with a per node Softmax activation on the output of the last layer, with the objective of minimizing a cross-entropy loss on all the labeled nodes through gradient descent backpropagation.

An overview of the node classification model with a two-layer RGCN can be observed in Fig.1 [11].

Fig. 1. Node Classification RGCN. Different color is used to highlight different entity types, unlabelled entities are in grey. Figure taken from [11].



2.3 Explainability on Graph Neural Networks

Graph Neural Networks (GNNs) are becoming increasingly popular, and at the same time the subfield of explainability in GNNs is getting more and more

attention. Despite the progress and effectiveness of different kinds of GNNs, the rationales of predictions of those models are not easy for humans to understand. Unlike traditional image and text data, GNNs aggregate both node features and graph topology to make predictions, necessitating the uncovering of important subgraphs and sets of features, referred to as explanations, to comprehend their decision-making process.

The explanation methods commonly used for image and text data often rely on grid-like structures with fixed neighbors, leveraging locality information. However, graphs do not have predefined grids, and each node may have a varying number of neighbors, with topology represented through adjacency matrices. As a result, adapting explanation methods from images to graphs requires innovative approaches that can account for the unique graph structure.

A fundamental aspect of explainability for graph data is feature importance. In traditional image data, feature importance can be treated as a trainable variable optimized through back-propagation. However, graphs demand discrete representations due to the non-differentiability of the adjacency matrix, posing unique challenges for determining feature importance.

In [8], a taxonomy of existing GNNs explanations approaches is proposed, where the instance-level explanations approaches are grouped into: Gradients, Features, Perturbations, Decompositions, and Surrogates.

Considering the research question, which involves an investigation of knowledge injection and various initialization methods for perturbation-based approaches, the primary focus will be on existing perturbation-based explanation methods to explain GNNs. These methods utilize gradients to assess the importance of different input features and monitor how predictions change with different input perturbations to derive input importance scores.

To explain deep graph models, perturbation-based methods generate masks to highlight essential input features. Depending on the explanation task, different masks are generated such as node masks, edge masks and node feature masks. These masks are then combined with the input graph, creating a new graph that incorporates crucial feature information. This modified graph is fed into the trained GNN to evaluate the masks and refine the mask generation algorithms. The goal is to ensure that the masks capture important features that convey key semantic meaning and lead to predictions similar to the original model's output.

The main differences between perturbation-based methods lie in three aspects: the mask generation algorithm, the type of masks and the objective function [8].

There are three different types of masks: soft masks, discrete masks, and approximate discrete masks.

- **Soft Masks**: These masks consist of continuous values between 0 and 1 and can be updated directly through back-propagation. However, they suffer from the "introduced evidence problem." Any non-zero or non-one value in the mask may introduce new semantic meaning or noise to the input graph, thereby affecting the explanation results.

- **Discrete Masks**: In contrast to soft masks, discrete masks only contain binary values (0 and 1), avoiding the introduced evidence problem. However, they involve non-differentiable operations.
- **Approximate Discrete Masks**: These masks use reparametrization tricks and sparse relaxations to approximate discrete masks. While the output mask is not strictly discrete, it provides a good approximation that enables back-propagation, alleviating the introduced evidence problem.

The main perturbation-based methods include GNNExplainer [10], PGExplainer [15], GraphMask [16], Zorro [17], Causal Screening [18] and SubgraphX [19].

As this study primarily builds upon the extension of GNNExplainer, which served as the foundation for all the other perturbation-based techniques mentioned, only this particular method will be discussed in depth.

2.4 GNNExplainer

GNNExplainer is a model agnostic explainability tool designed specifically for Graph Neural Networks (GNNs). Introduced in a research paper by Ying et al. in 2019 [10], GNNExplainer utilizes a perturbation-based approach to offer insights into GNN predictions by identifying the most influential edges and node features for a given node.

The primary objective of GNNExplainer is to determine which edges and node features contribute the most to the model's prediction on a specific node within the graph. To achieve this, GNNExplainer learns soft masks, representing the importance scores of edges and node features, to provide explanations for the model's decisions.

The process of obtaining these masks begins with random initialization, treating the soft masks as trainable variables. These soft masks are then combined with the original graph using element-wise multiplications. Next, the masks undergo optimization through a process that involves maximizing the mutual information between the predictions made by the original graph and the predictions obtained from the newly modified graph: this optimization process is explained in more detail in Section 2.4.

One of the key features of GNNExplainer is its ability to provide local explanations for each input graph individually. This means that the method outlines important edges and features specific to each node, as different masks are learned for different nodes. As a result, GNNExplainer can offer local node-level explanations, highlighting the importance of edges and features for each node's prediction.

GNNExplainer objective function In GNNExplainer the objective function is derived from a formalization of the notion of importance using mutual information MI - which quantifies the change in the probability of the prediction of node v, when its computation subgraph is limited to explanation subgraph G_s.

The optimization framework is defined as follow:

$$\max_{G_s} MI(Y, (G_s, X_s)) = H(Y) - H(Y|G = G_s, X = X_s) \tag{2}$$

Due to the fact that $H(Y)$ is constant and fixed for a trained model, Eq. 2 can be rewritten as a minimization of the conditional entropy: in that sense GNNExplainer explanation for prediction of node v is a subgraph G_s which minimizes the uncertainty of the model when the input graph is limited to G_s .

$$H(Y|G = G_s, X = X_s) = -E_{Y|G_s,X_s}[\log P(Y|G = G_s, X = Xs)] \tag{3}$$

Direct optimization of GNNExplainer is not tractable as there is an exponential number of subgraphs that are candidate explanations for the prediction on node v. For that reason GNNExplainer introduces a continuous relaxation - in the form of subgraph constraint - which can be interpreted as a variational approximation of the distributions of subgraphs of the original graph. Furthermore, by considering G_s, the subgraph explanation, as a random graph variable, an upper bound can be found for the objective function 3 through Jensen's inequality:

$$min_G H(Y|G = E_G[G_s], X = X_s) \tag{4}$$

Due to the complexity of neural networks, the convexity assumption doesn't hold and thus it is not assured that a global minimum can be found - but experimentally it has been found that minimizing this objective with regularization often leads to local minimum corresponding to high-quality explanations.

In the optimization framework defined in Eq. 4, $E_g[G_s]$ is replaced by an optimized mask of the adjacency matrix, as the element-wise multiplication between the adjacency matrix and the sigmoid of the mask parameter - which is the parameter to be optimized. In particular, the conditional entropy defined in Eq. 4 can be modified with a cross entropy objective between the label class and the explanation prediction - which is the prediction obtained with the masked subgraph - and can be overall thought as a prediction loss component and be written like this:

$$\text{prediction loss} = \sum_{c=1}^{C} 1[y = c]P(Y = y|G = A_c * \sigma(M), X = X_c) \tag{5}$$

GNNExplainer's objective also include regularizations objectives such as L1 lasso regularization to penalize the size of the explanation, and element-wise entropy to encourage the structural mask to be discrete, which can be respectively written as follows:

$$\text{size loss} = \sum |\sigma(M)| \tag{6}$$

$$\text{entropy loss} = -\sigma(\log(M)) - (1 - \sigma(M)) * \log(1 - \sigma(M)) \tag{7}$$

As will be discussed in Section 3.1, the objective function of GNNExplainer will be adjusted according to the characteristics of the RGCN model, in order to allow for an accurate and sparse explanations.

2.5 Semantic Explanations of Deep Learning models

Semantic Web technologies offer semantically interpretable tools which allow reasoning over knowledge bases that, as stated in [9, 20], have the potential to facilitate explanations in Machine Learning Systems. From a general point of view, Semantic Web technologies have primarily been used to make two types of ML models explainable: supervised classification tasks using Neural Networks and unsupervised embedding tasks [20].

Regarding combining classification tasks with Semantic Web Technologies, one approach is to map network inputs or neurons to classes of an ontology or entities of a Knowledge Graph. For example in [21], image contents are extracted as RDF triples and then matched to DBpedia via the predicate `owl:SameAs`. Through the mapping between the input content to KGs, both the results and the functioning of the network can indeed become more understandable for humans.

Another growing approach includes methods to fuse deep learning representations with expert knowledge graphs, as the X-NeSyL method proposed in [22]. This approach involves the concrete use of two notions of explanations, at inference and training time. In particular, EXPLANet is a compositional CNN that makes use of symbolic representations and SHAP-Backprop is an explainable AI-informed training procedure that corrects and guides the DL process to align with symbolic representations in the form of KGs.

While these approaches show potential in explaining deep learning models using Semantic Web technologies, they require establishing a connection between entities and external KGs. As a result, they may not be tailored to models that directly take in the entire structure of a KG, like RGCN.

Conversely, the RGCNExplainer implementation will utilize the advantage of these approaches, which lies in utilizing the semantic context embedded in the mapped entities.

3 Methodology

In this section, we delve into the implementation of RGCNExplainer, explore the reasoning behind incorporating prior knowledge into explanations, and provide an overview of the Relation Attribution Approaches.

3.1 RGCNExplainer: Expanding GNNExplainer for RGCNs

Expanding GNNExplainer to work with RGCNs requires adapting the explanation generation process to accommodate the unique characteristics of RGCNs, particularly their operation on relational graphs where entities are connected by different types of relationships.

In RGCN, each node in the graph represents an entity, and edges represent relationships between entities, annotated with their respective edge types. The primary difference between traditional graphs and KGs, like those used in RGCN, lies in the representation of the adjacency matrix. In non-heterogeneous

graphs, the adjacency matrix is a single square matrix with dimensions n × n (where n is the number of nodes). However, in heterogeneous graphs, such as KGs, the adjacency matrix consists of multiple adjacency matrices stacked vertically or horizontally, with each matrix representing a specific relation type, as can observed in Fig. 2. This leads to sparsity in the adjacency matrix, necessitating the use of a sparse representation for efficient convolutions on such graphs during training of RGCN and RGCNExplainer.

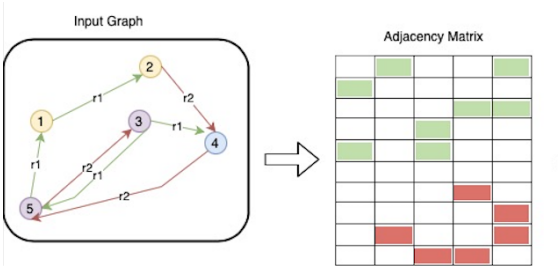


Fig. 2. Relational Graph into Stacked Adjacency matrices. Each edge is represented as a colored box in the Adjacency matrix: green and red boxes represent type 1 and type 2 relations, respectively.

Additionally, the distribution of relations in multi-relational data must be considered to prevent the model from overfitting to the most frequent relations. RGCN tackles this issue by introducing a relation-based regularization in its objective function, together with a condition on the most frequent relation present in the neighborhood of the node to be explained.

Moreover, to enhance the sparsity and faithfulness of the explanations obtained with RGCNExplainer, an additional step is incorporated compared to the GNNExplainer approach. The optimization task of RGCNExplainer produces a soft mask that represents the importance weights assigned to each edge in the neighborhood of the target node to be explained.

To generate a concise explanation, the process begins with a minimum explanation size, after which the most crucial edges are selected as explanation edges until the subgraph explanation matches the model's prediction being explained. Alternatively, the process stops when the maximum explanation size is reached. This iterative selection of edges ensures that the resulting explanation reaches the highest possible faithfulness while maintaining a considerable level of sparsity.

RGCNExplainer Pipeline The RGCNExplainer pipeline thus includes the steps described in Algorithm 1, and discussed below.

RGCNExplainer takes as input the node to be explained v, the KG to which node v belongs, the trained model f, the prediction of the model on node v y, and constraints on minimum and maximum explanation size.

Initiating the process, the input KG is transformed into a Sparse Adjacency matrix format, as depicted in Fig. 3. Subsequently, the mask parameter is initialized based on either a baseline or prior knowledge initialization. This mask parameter is then fine-tuned through backpropagation to assign importance weights to edges, optimizing the objective function outlined in the following section. The training loop concludes by selecting the k most important edges - through a threshold operation -, where k represents the minimum explanation size. Additional crucial edges are successively incorporated until the generated explanation subgraph produces the same prediction as the full subgraph, or until the maximum explanation size is attained.

Algorithm 1: RGCNExplainer: Given a node $v = (Av)$ where $f(v) = y$, generate the minimal subgraph explanation such that $f(M \cdot Av) = y$

Data: node $v = (Av)$, trained RGCN model f, RGCN predictions $y = f(v)$, KG, min size explanation e, max size explanation E, SUB_HOR_VER: get the second hop neighbor vertically and horizontally stacked Adjacency matrices, THR: select edges according to a specified threshold

Result: Explanation for node v in terms of masked adjacency matrices

1. $sub\_hor\_graph\_orig$, $sub\_ver\_graph\_orig \leftarrow$ SUB_HOR_VER(KG)
2. $\hat{M} \leftarrow J_n$ # *Mask initialization*
3. **for** K *iterations* **do**
   - $sub\_hor\_graph \leftarrow \sigma(sub\_hor\_graph\_orig) \cdot \hat{M}$;
   - $sub\_ver\_graph \leftarrow \sigma(sub\_ver\_graph\_orig) \cdot \hat{M}$;
   - $y\_pred \leftarrow f(sub\_hor\_graph, sub\_ver\_graph)$ # *Perturbed forward*;
   - $L \leftarrow L(y\_pred, y, v, f)$ # *Loss* ;
   - $\hat{M} \leftarrow \hat{M} + \alpha \cdot \Delta_{\hat{M}} L$ # *Mask update* ;

   end
4. $y\_t \leftarrow f(\text{THR}(sub\_hor\_graph, e), \text{THR}(sub\_ver\_graph, e))$;
5. **while** $y\_t \neq y$ **do**
   - $y\_t \leftarrow f(\text{THR}(sub\_hor\_graph, e + i), \text{THR}(sub\_ver\_graph, e + i))$;
   - i += 1;
   - **if** $i > E$ **then**
     - **break**;

   end

   end

Objective Function The Design of the Objective function for RGCNExplainer is directly inspired by GNNExplainer as described in Section 2.4. Still, due to characteristics specific to the RGCN model, some modifications have been implemented.

Two loss components have been introduced, aiming to balance fidelity and compactness in the explanations.

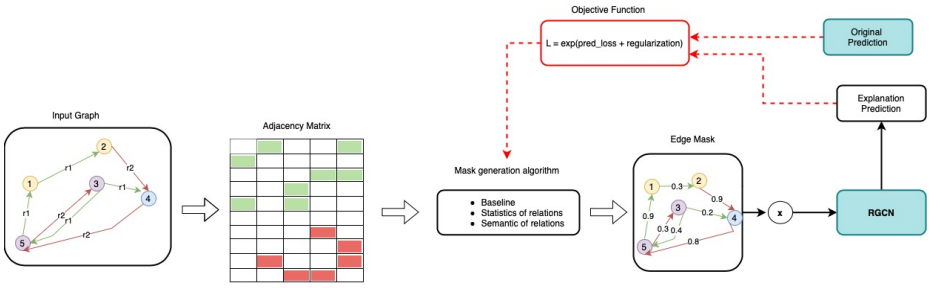


Fig. 3. RGCNExplainer pipeline.

The *Standard Deviation Size Loss* seeks to maximize the standard deviation of the mask values, pushing the mask values to become more skewed, resembling a discretized mask with values closer to either 0 or 1. This is relevant because RGCNExplainer learns a soft mask, whilst combining the binarized version of that soft mask with the input graph, where the binarization happens via a thresholding at 0.5.

$$\text{Standard deviation size loss} = \sum_{i=1}^{n} \frac{(Mi - \bar{M})^2}{n} \tag{8}$$

In order to address the distribution of relations and avoid overfitting to the most frequent relations, the *Most Frequent Relation Specific Loss* has been introduced. This loss takes into account the prevalence of relations in the node subgraph. The model first identifies the most frequent relation in the node subgraph by mapping the indices to the original relation set. Then, the loss is defined as the ratio between the number of important edges associated with the *Most Frequent Relation* and the total number of important edges in the subgraph. The objective is to minimize this value during training. By incorporating this loss, the model is guided to be less reliant on the most frequent relation and more attentive to rare relations in the subgraph. This helps prevent overfitting and improves the generalization capabilities of the model in handling different relation types.

$$\text{Most frequent relation specific (r*) loss } = \frac{|M_r* > threshold|}{|M > threshold|} \tag{9}$$

It must be mentioned that the inclusion of the *Standard Deviation Size Loss* transforms the objective of RGCNExplainer into a min/max objective. To ensure the loss remains non-negative, the exponential of the overall loss is considered.

The complexity of the loss function, which involves components to be both minimized and maximized, necessitates determining appropriate loss coefficients. These coefficients directly influence the weight of each loss component in the overall loss and, consequently, the overall performance of RGCNExplainer. A

more detailed discussion about how those hyperparameters have been optimized can be found in Section 4.1.

It is essential to note that no ground-truth explanations exist for RGCNExplainer, and thus defining the ideal balance between the different optimization objectives is not straightforward, given the absence of a clear reference for evaluation.

Knowledge Injection via Mask Initialization Being perturbation-based explanation methods, both GNNExplainer and RGCNExplainer are significantly affected by their mask initialization. Thus, conducting experiments with various mask initializations becomes a valuable approach to incorporate prior domain or statistical knowledge into the explanations. By doing so, it becomes possible to bias the explanations towards specific behaviors, guiding the interpretability process in a desired direction.

In particular, the following mask initializations are introduced and experimented with:

1. **Baseline - from GNNExplainer**
   (a) *Normal*: The mask weights are initialized according to a normal distribution with mean equal to 1 and standard deviation equal to a scaled version of the Relu gain: $N(1, \sqrt{2} * \sqrt{\frac{2}{2*numnodes}})$
2. **Prior Statistical Knowledge**
   (a) *Overall Frequency* : The initialization weight w_r for relation r is calculated as the ratio between the number of times the relation r occurred in the dataset (denoted by n_r) and the total number of triples in the dataset (denoted by N_t): $w_r = \frac{n_r}{N_t}$. This approach biases the mask initialization towards the most frequent relations, assigning higher weights to the edges of these relations. It thus helps in aligning the explanations with the distribution of relations in the dataset and incorporating statistical knowledge into the explanation process.
   (b) *Relative Frequency*: The mask weights are initialized per relation, taking into account the frequency of the relation in the neighborhood of the node to be explained. Let w_r represent the initialization weight for relation r, which is computed locally and depends on the node. The weight w_r is calculated as the ratio between the number of times the relation r occurs in the neighborhood of the node (denoted by $n_r g$) and the total number of relation occurrences in that neighborhood (denoted by $N_t g$): $w_r = \frac{n_r g}{N_t g}$. This approach allows the explanation process to be influenced by the local distribution of relations around the node being explained, giving more importance to relations that are prevalent in that specific neighborhood.
   (c) *Inverse Relative Frequency*: Similar to the previous method, the mask weights are initialized per relation and are node-dependent. However, this time, the weight w_r is computed as 1 minus the frequency of the relation in the node's neighborhood: $w_r = 1 - \frac{n_r g}{N_t g}$. By using the inverse

relative frequency, the explanation model emphasizes relations that are less common in the neighborhood of the node, encouraging a focus on rare relations and potentially uncovering hidden patterns in the data.

(d) *Domain and Range Frequency*: The mask weights are initialized per relation, based on the number of distinct object classes they have as either domain or range. In this initialization approach, each relation's weight, denoted as w_r, is determined by considering the diversity of object classes associated with that relation. Specifically, if a relation has a larger number of different object classes as either the domain or the range, it will receive a higher weight during initialization. This method aims to bias the mask towards relations that have a broader scope and involve a more diverse set of object classes. By doing so, the explanation process is guided to prioritize relations that play a more significant role in connecting various object types, potentially leading to more informative and insightful explanations.

3. **Prior Domain Knowledge**

(a) *Domain Knowledge*: In order to inject prior domain knowledge into the explanation, the mask can be initialized by giving different specified weights to different relations. This approach enables researchers or domain experts to direct the explanation process towards areas of interest, helping them gain deeper insights and understanding from the model's explanations.

Technical implementation: RGCN Model The proposed approach aims to provide explanations for the predictions generated by a node classification RGCN model. The model takes horizontally and vertically stacked adjacency matrices as input, which represent the underlying KG. This input format is chosen because, during the training of RGCNExplainer, the model predicts the class of the node to be explained. The prediction is made by using updated masked adjacency matrices, where the mask serves as the parameter of interest in the explanation process.

The RGCN model implementation utilized in this approach is described in [23]. It comprises two RGCN layers that take a one-hot encoding of the nodes as input. The one-hot encoded input is first mapped to a hidden layer, and subsequently to class probabilities for prediction. The training of RGCN models is performed using a full-batch approach for 50 epochs. The hyperparameters used during training include a hidden size of 16, ReLU activation function, Adam optimizer with a learning rate of 0.01, and an L2 penalty of 0.5×10^{-3} applied to the weights of the first layer.

3.2 Metrics

To quantitatively evaluate the explanations provided by RGCNExplainer, several metrics are considered:

Fidelity Fidelity assesses the faithfulness of the explanations to the model's decision-making process. In other words, the explanations should accurately identify the important input features that influence the model's predictions.

To measure Fidelity, the metrics Fidelity+ and Fidelity- are introduced in [8]. Fidelity+ measures the impact of removing important input features on the model's predictions. It is based on the assumption that important features are discriminative, and when they are removed, the model's predictions should change significantly. On the other hand, Fidelity- evaluates the predictions when only the important features are retained, and predictions should remain stable.

Formally, let G_i represent the i-th input graph, and $f(\cdot)$ be the RGCN classifier to be explained, where $\hat{y}_i = \text{argmax}\, f(G_i)$ is the predicted class. The explanations provided by RGCNExplainer are represented as discrete mask terms, denoted as m_i, where each element is either 0 or 1, indicating the importance of the corresponding feature.

Fidelity+ is defined as the difference between the accuracy or predicted probability of the original predictions and the explanation predictions after masking out the important input features, where higher values of Fidelity+ indicate more accurate and reliable explanations:

$$\text{Fidelity+} = \frac{1}{N}\sum_{i=1}^{N}(1(\hat{y}_i = y_i) - 1(\hat{y_i}^{1-mi} = y_i)) \tag{10}$$

On the contrary, Fidelity- examines the prediction change when important features are removed. In this context, important features are expected to carry discriminative information that contributes significantly to the model's predictions. Thus, even in the absence of unimportant features, the presence of important features should lead to predictions that remain similar to the original.

Formally, Fidelity- is defined as follows:

$$\text{Fidelity-} = 1 - \frac{1}{N}\sum_{i=1}^{N}(1(\hat{y}_i = y_i) - 1(\hat{y_i}^{mi} = y_i)) \tag{11}$$

It is worth noting that in the original paper [8], Fidelity- is defined in positive terms. However, to enhance the readability of the tables and ensure that high values for all metrics indicate better explanations, the metric is adjusted by introducing a 1- term.

Sparsity Good explanations should exhibit sparsity, meaning they capture only the most important input features while disregarding irrelevant ones. The sparsity metric quantifies this property by measuring the proportion of selected important features, denoted as m_i, over all features, represented by M_i. Higher values of sparsity indicate more compact and focused explanations, which tend to be more informative and interpretable.

The sparsity metric is defined as follows:

$$\text{Sparsity} = \frac{1}{N}\sum_{i=1}^{N}(1 - \frac{m_i}{Mi}) \tag{12}$$

3.3 Relation Attribution Approaches

To investigate the influence of different relations on the models' predictions, two relation attribution methods inspired by forward feature selection and backward elimination have been implemented.

Forward Relation Selection Forward feature selection typically involves iteratively adding one variable at a time to the model while observing changes in prediction or some other criterion [24, 25]. However, in our case, we are interested in studying the impact of different relations on predictions individually. Therefore, at each iteration, we analyze the effect of a specific relation on the model's prediction.

The explanation approach takes the trained RGCN model as input, and for each node whose prediction is to be explained, the edges of only one relation type are retained at each iteration. Given the representation of the graph data as an adjacency matrix, this means that at each iteration, only the edges of the relation of interest have a value of 1, while all other entries are set to zero. Consequently, the modified adjacency matrix is used as input for a forward pass on the trained RGCN model. In particular, due to how RGCN is structured - the adjacency matrix is represented in terms of horizontally and vertically stacked matrices where each matrix stores the edges specific to each relation.

If the prediction made on the graph containing only one relation at a time is correct, the corresponding relation is considered important based on the forward relation selection approach. In other words, relations that individually influence the model's predictions and yield accurate results when isolated are regarded as significant according to this method.

Backward relation elimination Backward feature elimination can be seen as the opposite of forward selection. Instead of adding a feature at each iteration, at each step, a feature is removed, and the model's behavior is analyzed accordingly [24, 25].

Similar to how we defined forward relation selection, backward relation elimination is inspired by backward elimination but is inherently different. In backward relation elimination, a relation is considered important if, upon removing the edges of that relation type from the adjacency matrices and using the modified graph for a forward pass, the model's prediction is no longer correct. This approach can be thought of as counterfactual, as it explores how the model's predictions change when specific relations are eliminated.

As with forward relation selection, backward relation elimination also examines the impact of relations on predictions individually. At each iteration, one relation is removed, and its effects on the model's predictions are studied independently. Relations that, when removed, lead to incorrect predictions are considered important according to this method.

4 Experiments

This section describes the experimental setup and results to address the Research Questions presented in Section 1, which involved the implementation of RGCNExplainer and the exploration of the impact of knowledge injection on RGCN explanations (RQ 1) and the investigation of the relations that define a class through a statistical analysis of the RGCN explanations and the implementation of the Relation Attribution method (RQ 2).

4.1 Experimental Setup

The experimental setup includes the introduction of the datasets and a discussion about hyperparameter search.

Datasets. Experiments were conducted on a total of four datasets, consisting of two small benchmark datasets and two larger datasets more representative of real-world scenarios.

The datasets were converted to the format proposed in [23], where a collection of KG datasets is introduced for evaluating Relational Machine Learning tasks. The original RDF format was transformed into a set of CSV files, where all relations and nodes were mapped to integer indices. This conversion simplifies data reading into any data science or machine learning software without the need to parse RDF or load the data into a triple store.

It must be noted that in these datasets relations do not necessarily encode directed subject-object relations, but are also used to encode the presence, or absence, of a specific feature for a given entity. In each dataset, the targets to be classified are properties of a group of entities represented as nodes.

- The **AIFB** dataset is a Semantic Web (RDF) dataset that describes the AIFB research institute (Applied Informatics and Formal Description Methods) at the University of Karlsruhe in terms of its staff, research group, and publications. The goal is to predict the affiliation (research group) for the people in the dataset. [6]
- The **DBO GENDER** dataset is introduced in this work as a subgraph of the DBPedia Knowledge Graph, built with the task of predicting the gender of nodes representing People. It was constructed via a SPARQL query through the Triply SPARQL endpoint, encompassing information about 1000 people balanced in gender.
- The **AMPLUS** (Amsterdam Museum) dataset is dedicated to the history of Amsterdam, whose content has been translated to Linked Open Data. The task is to predict the type of a given collection item. The AMPLUS dataset has been introduced in [23] as a modification of the original AM dataset, where only a subset of the relations have been included and categories are remapped to a smaller set of classes to create a more balanced class distribution.

– The **MDGENRE** dataset was presented in [23] as a collection of movies that have either won awards or been nominated. This dataset forms a KG primarily focused on movie-related information. The longest path within this graph is of length 4. Notably, each movie in this dataset has been associated with a single genre. This specific setup is designed for the purpose of predicting the genre of each movie.

Table 1. Dataset Statistics. Values* have been computed on a stratified random sample of nodes.

Metrics	AIFB	DBO GENDER	AMPLUS	MDGENRE
# Nodes	8285	11534	1 153 679	349 344
# Classes	4	2	8	12
# Relations	45	225	33	154
# Triples	26666	15029	2 521 046	1252247
Training Set	103	466	13 423	3846
Validation Set	35	251	20 000	1006
Avg node degree	8.086 ± 31	2.606 ± 6	4.37	7.169 ± 156
Avg node degree (2 hops)	471.484 ± 605	43.298 ± 93	2.0	2.0
Avg val node degree	26.429 ± 42	16.394 ± 9	9.312 ± 3 *	32.062 ± 21 *
Avg val node degree (2 hops)	1380.429 ± 386	213.793 ± 155	73472.344 ± 24 *	24331.021 ± 9958 *
Validation Accuracy	1	0.55	0.59	0.57

Table 2. Set of hyper-parameters over which the grid search was performed. The values in bold represented the chosen set.

Lr	Weight Decay	Entropy	Most freq Relation (mfR)	Prediction	Size	Threshold
0.1,**0.5**	0.1, **0.9**	**1**,10	**1**,10	**1**,10	**0.0005**,0.005	0.5

Hyper-parameter Search. The RGCNExplainer implementation involves training a separate model for each node to be explained, making it impractical to find optimal hyperparameters for each individual model. Assuming that a single set of hyperparameters that works well for one node will perform equally well for all other nodes is a strong assumption and requires validation.

To assess the validity of this assumption, several experiments were conducted to evaluate whether using the same hyperparameter set could lead to satisfactory results across different nodes and mask initializations. The hyperparameters considered for evaluation include the learning rate, weight decay, and coefficients for various components of the RGCNExplainer objective function. To gauge the quality of explanations during the grid search process, a scoring metric was devised by summing the values of Sparsity, Fidelity+, and Fidelity-.

To test the hypothesis that a common set of hyperparameters could be optimal for different nodes and mask initializations, multiple grid searches were performed on the benchmark AIFB dataset, considering nodes from various classes

and different mask initializations. The results demonstrated that the same hyperparameter set did not consistently yield the best performance across these experiments. Nonetheless, through experimentation, a set of parameters that exhibited good performance across nodes and mask initializations was identified and used in the subsequent experiments, which is reported in Table 2.

Moreover, through experimentation, it was discovered that omitting the most prevalent relation in the neighborhood— if accounting for more than 70% of total edges— resulted in accelerated selection of pivotal edges and an overall improvement in explanation performance.

Indeed, in a relational context, the size of the second-hop neighborhood is often notably larger compared to the first-order neighborhood of a node. In many cases, this discrepancy arises from the prolific occurrence of a specific relation, frequently identified as the `rdf:type` relation.

4.2 Answering RQ 1: RGCNExplainer and knowledge injection in the explanations

RGCNExplainer provides localized explanations for RGCN predictions in node classification tasks. These explanations pinpoint the most pivotal edges from the model's perspective. To convey and document these localized explanations, they are visualized through selected subgraphs. An illustrative visualization is provided for each dataset.

To probe the impact of injecting knowledge into RGCNExplainer results, a series of experiments were conducted using various mask initializations. For each dataset, a stratified random sample of validation nodes per class was chosen. RGCNExplainer was then employed with different mask initializations to generate corresponding subgraph explanations.

The metrics Fidelity-, Fidelity+, and Sparsity were computed and reported for explanations derived from each mask initialization. These metrics were also compared against metrics computed on randomly generated baseline explanation subgraphs. Higher metric values indicate better explanations.

The diverse mask initializations encompass the injection of prior statistical knowledge and Domain Knowledge. Notably, as Domain Knowledge was often unavailable for most benchmark datasets, it was derived from the outcomes of the Relation Attribution approach. This approach facilitated distinct initialization of explanations for nodes within different classes, based on significant relations identified using both forward and backward relation attribution methods.

Results.

AIFB. In Fig. 4, a representative visualization of an explanation for a randomly chosen node is presented. It's noticeable that RGCN Explainer has chosen a subgraph from the second-level neighborhood of the node. This subgraph is deemed significant as it contains the most crucial edges according to the method's criteria.

Fig. 4. AIFB: Example Explanation Visualization. The node in red is the node whose explanation subgraph is reported.

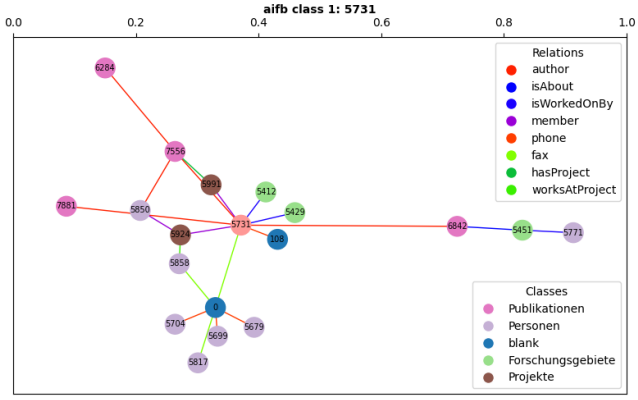


Table 3. AIFB: Results on RGCNExplainer with different mask initializations - *init*. For each experiment the metrics on Sparsity and Fidelities are reported together with a Score which is defined as a sum of those metrics.

init	Score	Sparsity	Fidelity-	Fidelity+	Fidelity- random	Fidelity+ random
Normal	**2.537**	0.966	**0.971**	0.6	0.571	0.057
Overall Frequency	2.364	0.935	0.829	0.6	0.571	0.057
Relative Frequency	2.360	0.931	0.829	0.6	0.571	0.057
Inverse Relative Frequency	2.364	0.935	0.829	0.6	0.571	0.057
Domain Frequency	2.363	0.934	0.829	0.6	0.571	0.057
Range Frequency	2.362	0.933	0.829	0.6	0.571	0.057
Domain Knowledge Forward	2.507	0.964	0.943	0.6	0.543	0.057
Domain Knowledge Backward	2.513	**0.970**	0.943	0.6	0.486	0.057

For the AIFB dataset Table 3 presents the results of RGCNExplainer with different mask initializations. Notably, both Sparsity and Fidelity- metrics consistently display high values across all initializations, indicating that RGCNExplainer generates accurate and concise explanations. The choice of mask initialization significantly impacts the method's performance, particularly in terms of the compactness of subgraph explanations.

For explanations on the AIFB dataset, the Normal mask initialization demonstrates the best overall performance, closely followed by the Domain Knowledge Initialization, which yields slightly weaker results.

DBO GENDER. In Fig. 5, a representative visualization of an explanation for a randomly chosen node is presented.

For the DBO GENDER dataset, Table 4 presents the results of RGCNExplainer with different mask initializations. Also for the explanations on the DBO GENDER dataset Sparsity and Fidelity- metrics consistently display high values across all initializations.

Contrarily to the AIFB dataset, the range frequency mask initialization demonstrates the best overall performance, closely followed by all the other ini-

Fig. 5. DBO GENDER: Example Explanation Visualization. The node in red is the node whose explanation subgraph is reported.

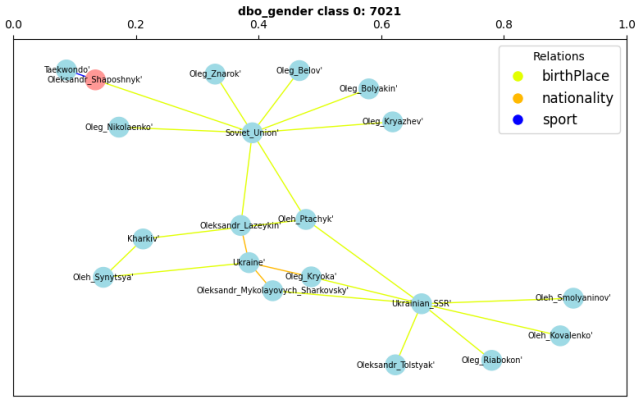


Table 4. DBO GENDER: Results on RGCNExplainer with different mask initializations - *init*. For each experiment the metrics on Sparsity and Fidelities are reported together with a Score which is defined as a sum of those metrics.

init	Score	Sparsity	Fidelity-	Fidelity+	Fidelity- random	Fidelity+ random
Normal	2.091	**0.731**	0.94	0.42	0.68	0.16
Overall Frequency	2.103	0.683	1.00	0.42	0.82	0.16
Relative Frequency	2.077	0.677	0.98	0.42	0.80	0.12
Inverse Relative Frequency	2.104	0.684	1.00	0.42	0.84	0.16
Domain Frequency	2.104	0.684	1.00	0.42	0.82	0.16
Range Frequency	**2.106**	0.686	1.00	0.42	0.84	0.16
Domain Knowledge Forward	2.097	0.697	0.98	0.42	0.72	0.14
Domain Knowledge Backward	2.097	0.697	0.98	0.42	0.72	0.14

tializations. This result is probably due to the underlying characteristics of the dataset: this could be due to the high number of relations in the dataset with high variability in terms of classes they have as range.

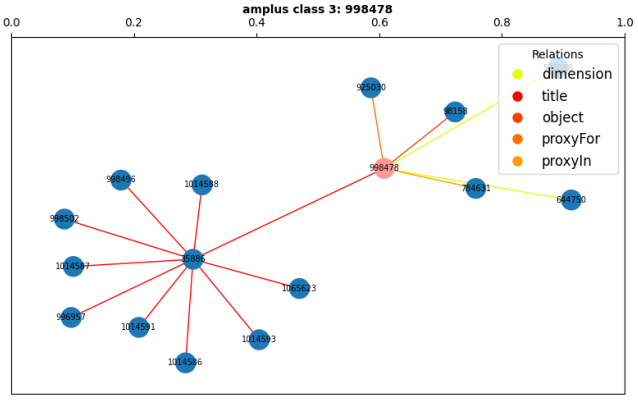
AMPLUS. In Fig. 6, a representative visualization of an explanation for a randomly chosen node is presented.

Due to computational constraints, particularly in generating explanation subgraphs for individual nodes within the AMPLUS dataset, extensive experiments involving all mask initializations were conducted solely on a single illustrative node. Providing an explanation subgraph for a single node in the AMPLUS dataset demanded an average of 20 minutes and 55GB of memory on a CPU.

The selected node - with id 998478 - belongs to the class of Drawings, and at the second hop neighborhood, it has 73454 neighbors and 73473 edges. Via those connections, RGCN learns to classify the node to the correct class of interest, and RGCNExplainer, given the RGCN prediction and the neighbors of the node, selects the edges that influenced most the RGCN decision to assign the selected node to the class of Drawings.

As it can be observed in Table 5 with all the mask initializations, the RGCN

Fig. 6. AMPLUS: Example Explanation Visualization. The node in red is the node whose explanation subgraph is reported.



Explanations have Fidelity- values equal to one. Thus predictions made with the trained RGCN using either the full or the explanation subgraphs output the same class. On the other hand, the level of sparsity varies across different explanations, meaning that the explanations have different sizes.

Table 5. AMPLUS: metrics on explaining the node 998478 from the class Drawings.

init	Score	Score Prob	Sparsity	Fidelity-	Fidelity+	Fidelity- random	Fidelity+ random
Normal	**2.99980**	2.23717	0.99980	1.0	1.0	0.0	0.0
Overall Frequency	2.99966	2.13636	0.99966	1.0	1.0	0.0	0.0
Relative Frequency	2.99966	2.13636	0.99966	1.0	1.0	0.0	0.0
Inverse Relative Frequency	2.99978	2.08110	0.99978	1.0	1.0	0.0	0.0
Domain Frequency	2.99966	2.13636	0.99966	1.0	1.0	0.0	0.0
Range Frequency	2.99966	2.13636	0.99966	1.0	1.0	0.0	0.0
Domain Knowledge Forward	2.99966	2.13636	0.99966	1.0	1.0	0.0	0.0
Domain Knowledge Backward	**2.99980**	2.23717	0.99980	1.0	1.0	0.0	0.0

MDGENRE. In Fig. 7, a representative visualization of an explanation for a randomly chosen node is presented.

Due to computational limitations, also on the MDGENRE dataset extensive experiments involving all mask initializations were conducted solely on a single illustrative node. Node 243773 was selected, which corresponds to the movie *Around the World in 80 Days* belonging to the Fantasy Film class. Within its second-hop neighborhood, this node interacts with 11,281 neighbors through 37,067 edges and RGCNExplainer managed to identify only the 87 most influential edges.

It is worth emphasizing that, for this particular node, RGCNExplainer provides an explanation with Fidelity- equal to one solely through Normal mask initialization, as can be observed in Table 6. Conversely, mask initializations

Fig. 7. MDGENRE: Visualization of explanation of node 243773 representing the movie *Around the World in 80 Days* to be classified as Fantasy Film.

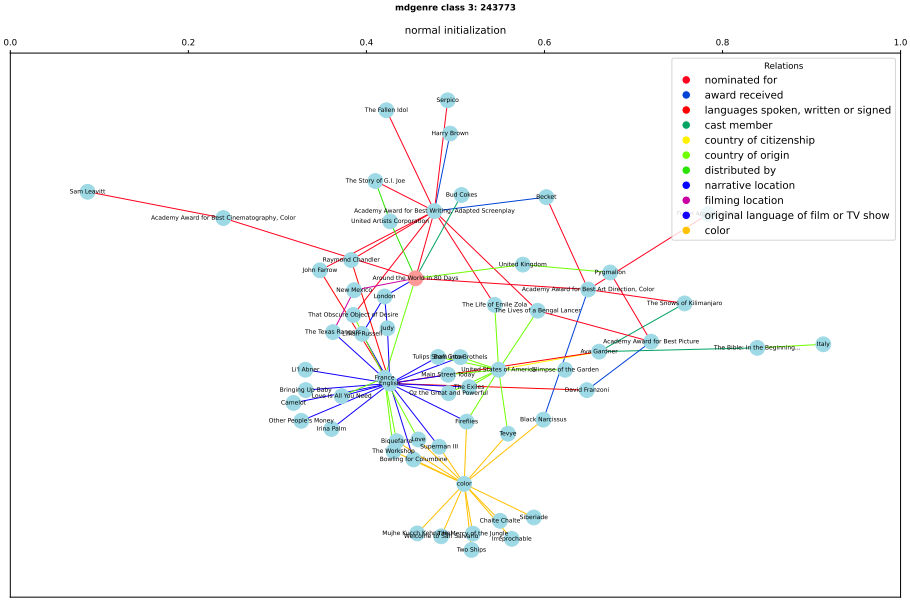


Table 6. MDGENRE: metrics on explaining the node for the movie *Around the World in 80 Days* (243773) from target class Fantasy Film (3).

init	Score	Score Prob	Sparsity	Fidelity-	Fidelity+	Fidelity- random	Fidelity+ random
Normal	**2.99873**	1.99643	0.99873	1.0	1.0	0.0	0.0
Overall Frequency	1.98608	1.98608	0.98608	0.0	1.0	1.0	0.0
Relative Frequency	1.98608	1.98608	0.98608	0.0	1.0	1.0	0.0
Inverse Relative Frequency	1.98608	1.98608	0.98608	0.0	1.0	1.0	0.0
Domain Frequency	1.98608	1.98608	0.98608	0.0	1.0	1.0	0.0
Range Frequency	2.00000	1.99739	1.00000	0.0	1.0	1.0	0.0

based on prior statistical knowledge of the graph result in RGCNExplainer generating an erroneous explanation subgraph. This unexpected outcome can be attributed to the dataset's substantial diversity of relation types, rendering relation frequency-based initializations less effective in this context.

Discussion on Knowledge Injection through Mask Initializations The experiments involving various mask initializations for RGCNExplainer indicate that, in the majority of cases (three out of four datasets), the most optimal performance is attained using the baseline Normal mask initialization. This outcome is closely followed by the Domain Knowledge Initialization. Essentially, this implies that incorporating prior statistical knowledge about the input graph into RGCNExplainer does not lead to enhanced performance.

4.3 Answering RQ 2: The relations that define a class

To explore the relations that characterize a class, two analyses were carried out: an analysis of the distribution of relations and an assessment of the outcomes of the Relation Attribution approaches.

Comparisons between the distribution of relations within the explanation subgraph (obtained with Normal mask initialization) and the full neighborhood subgraph were conducted. This analysis was executed on a per-class basis, with important edges being averaged for each class in both the full neighborhood and explanation subgraphs. The relative frequency of relations was then visually compared using a barplot representation, supplemented by two distinct statistical tests.

To assess whether the distribution of relations between the original and explanation subgraphs significantly differs, a paired t-test was employed, evaluating the node-wise divergence of the two distributions.

Additionally, to determine whether the distribution of relations varies across the classes of interest, a one-way ANOVA was employed for each feature. For features demonstrating a noteworthy distribution variance across classes, a Tukey's Honestly Significant Difference (HSD) test was conducted. This test assists in identifying specific relations associated with different classes based on RGCN-Explainer's conceptualization. Specifically, the Tukey's HSD test calculates the Honestly Significant Difference (HSD), representing the minimum mean difference between any pair of groups that is required for the difference to be statistically significant.

For both of these statistical tests, a significant difference in distribution was deemed at an alpha level of 0.05.

It's worth noting that the effectiveness of the second test is influenced by the dataset size and the number of labeled nodes, which greatly impact the generalizability and reliability of the results.

Results.

AIFB. A paired t-test was performed to compare the relation distribution between the original and the explanation subgraph. The only relation that did not show a significant difference was the 'editor' relation.

To compare the relation explanation distribution across different classes, a one-way ANOVA test was conducted, followed by a Tukey's HSD test.

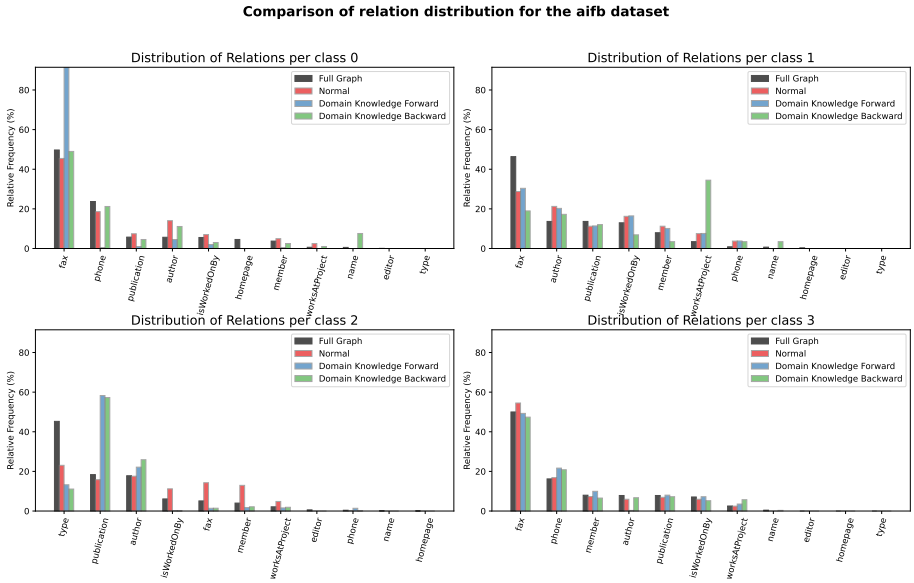
It was found that for relations 'phone' and 'fax' there is a significant difference between class 3 and all the others, whilst for relation 'photo' the difference was significant between classes 3, 0, and 2, whilst for relation 'member' only between class 3 and 0. This distinction is further evident in Fig. 8, where the relative frequency of relations per different classes is reported.

These findings offer valuable insights for attributing semantic meaning to the distinct classes of interest. Notably, due to the notable prevalence of relations like 'phone' and 'fax' in explanations for class 3, it's plausible to infer that class

3 encompasses agents within the AIFB research institute who engage in communications with external research institutes, given that these entities possess phone numbers and fax connections. Conversely, the prominence of the 'photo' relation in class 0 suggests that this class likely comprises individual agents with actual profile photos.

It is essential to note that the small size of the AIFB dataset and the limited size of each class of interest can make these results unreliable.

Fig. 8. AIFB: Comparison of Relation Distribution. Each subplot reports the relative frequency of the relations of interest comparing the frequency relation in the full subgraph and in selected explanation subgraphs.



For the AIFB dataset the relation attribution method demonstrated its ability to capture relations specific to different classes of interest in the AIFB dataset, as evident from Table 7. The proximity of the values to 1 indicates that these important relations were prevalent across all nodes within each class. Notably, the results displayed high-class specificity, indicating that the RGCN model effectively distinguishes and characterizes the various classes in the AIFB dataset by leveraging the semantic information embedded in the diverse relation types.

In particular it can be observed that relations 'fax' and 'name' are important for class 0, supporting our hypothesis that class 0 represent individual agents. In the same way the prevalence of relation 'publication' for class 2 suggests that the class represents PhD students or researchers who indeed have multiple publications.

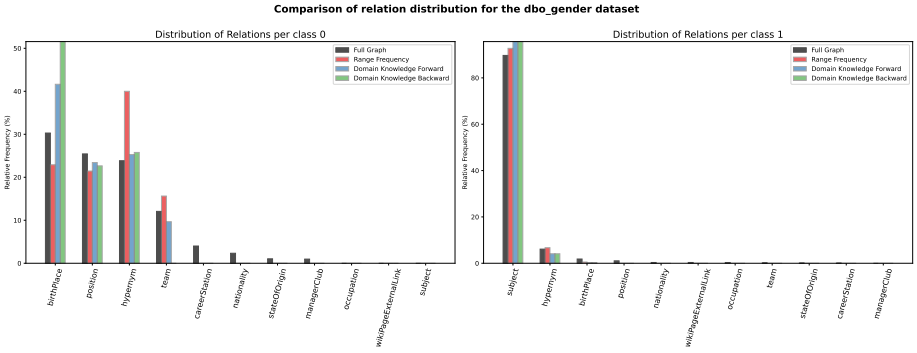
Table 7. AIFB: Relation Importance. For each class the percentage of nodes for which a relation was defined important through either a forward approach or backward approach is reported. Only if the relation was found to be important for more than 50 % of nodes for the given class the percentage is reported.

class	fax	homepage	name	phone	photo	publication	type	worksAtProject
0	1		1, 0.64				0.67	
1		0.6			0.9			0.8, 0.5
2				1		0.91, 1		
3							0.88	

DBO GENDER. The relation distribution analysis was conducted on the explanations obtained with Normal initialization. A paired t-test was performed to compare the relation distribution between the original and the explanation subgraph, showing no discernible variance in the statistical distribution for any relation.

Regarding the variations in distribution among explanations for nodes belonging to distinct target classes, specifically males and females, it's noteworthy to highlight a significant difference, established through the utilization of Turkey's Honest Significant Difference (HSD) test. It demonstrated a significant difference for the following relations: 'wikiPageExternalLink', 'subject', 'birthPlace', 'careerStation', 'team', and 'position'. This distinction is further evident in Fig. 9, where the relative frequency of each relation for each distinct explanation subgraph is graphically illustrated per class.

Fig. 9. DBO GENDER: Comparison of Relation Distribution. Each subplot reports the relative frequency of the relations of interest comparing the frequency relation in the full subgraph and in the selected explanation subgraphs.



Also for the DBO GENDER dataset the Relation Attribution approach shows that the RGCN model effectively distinguishes and characterizes the various classes, by leveraging the semantic information embedded in the diverse relation types. The results reported in Table 8 shows that for the class "Male" the impor-

Table 8. DBO GENDER : Relation Importance. For each class the percentage of nodes for which a relation was defined important through either a forward approach or backward approach is reported. Only if the relation was found to be important for more than 50 % of nodes for the given class the percentage is reported.

Label	birthPlace	careerStation	position	team	wikiPageExternalLink	subject	hypernym	wasDerivedFrom	isPrimaryTopicOf
Male	0.848	0.514	0.536	0.536			0.841	0.993	0.855
Female					0.531	1, 0.938			

tant relations include 'birthPlace', 'careerStation' , 'position', 'team', 'wikiPageExternalLink' , 'hypernym', 'wasDerivedFrom' and 'isPrimaryTopicOf'. Conversely, the 'subject' relation appears to be class-specific for the "Female" class.

AMPLUS. The relation distribution analysis was conducted on the explanations obtained with Normal initialization, and the corresponding barplot is reported in Fig. 10. A paired t-test was performed to compare the relation distribution between the original and the explanation subgraph: for all the relations the difference in distribution is significant at a 0.05 alpha level.

To compare the relation explanation distribution across different classes, a one-way ANOVA test was conducted, followed by a Tukey's HSD test. In particular the relation 'productionDateEnd' is significantly different for class "Photographs", whilst the relation 'maker' for class "Drawings".

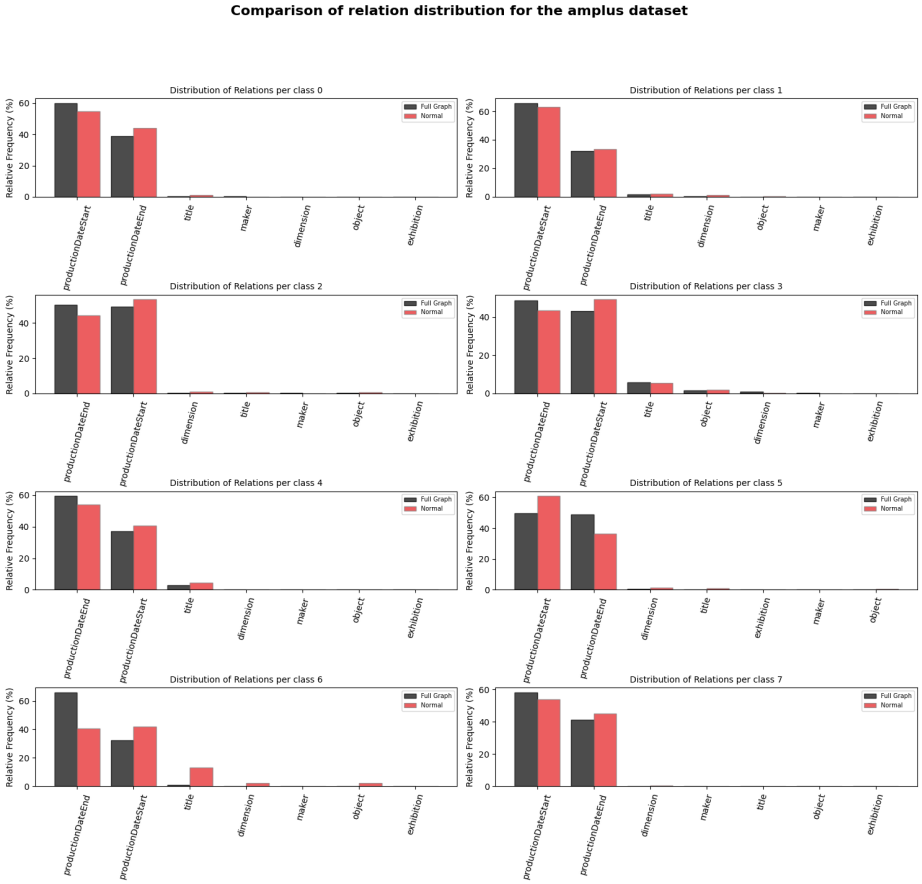
Table 9 displays the outcomes of the Relation Attribution method using the forward approach for the AMPLUS dataset. The remarkable class specificity observed in these results signifies the RGCN model's effectiveness in distinguishing and characterizing the various classes within the AMPLUS dataset. This achievement is primarily attributed to the model's ability to leverage the semantic information conveyed through diverse relation types.

Notably, the importance analysis revealed specific relations that significantly influenced certain classes. For instance, the 'maker' relation played a crucial role in the "Decorative Art" class, while 'title' was vital for "Metallic Art", and 'dimension' was significant for the "Book and Documents" class.

However, in contrast to the forward approach, the results obtained from the backward approach were less informative for the AMPLUS dataset. The backward approach tended to either identify almost no relations as important or showed no class specificity. Specifically, for the classes "Historical Artifacts" and "Paintings", all the relations were identified as important for all the nodes, while for the classes "Drawings" and "Books and Documents", none of the relations were identified as important.

MDGENRE. Fig. 11 reports the difference in relation distribution between the full subgraph and the normal explanation subgraph for the movie *Around the World in 80 Days* from the MDGENRE dataset . It can be noted that the major difference in relation distribution can be observed for relations 'country of origin' and 'original language of film or TV show', where the former is more

Fig. 10. AMPLUS: Comparison of Relation Distribution. Each subplot reports the relative frequency of the relations of interest comparing the frequency relation in the full subgraph and in the explanation subgraph.

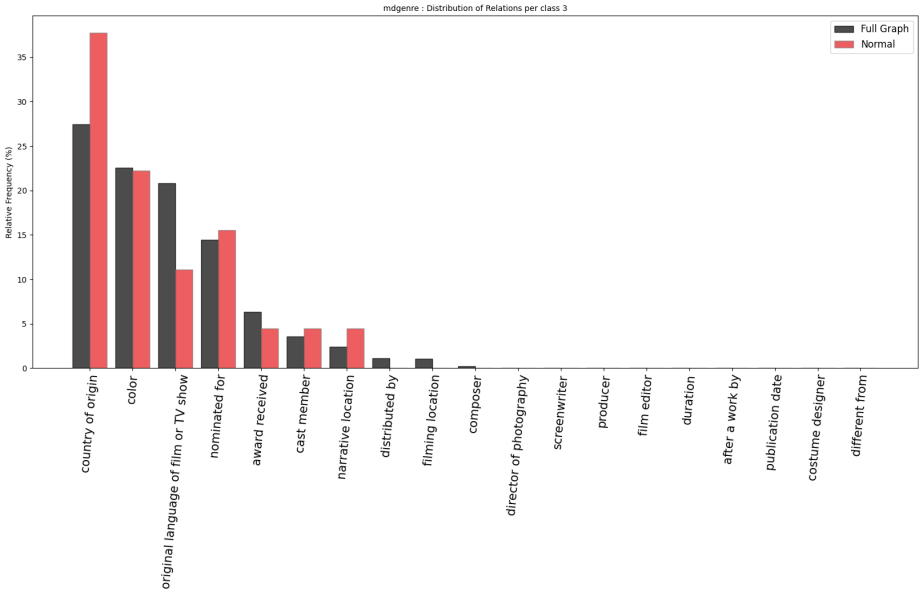


prevalent in the explanation, whilst the latter is more prevalent in the original subgraph.

Table 9. AMPLUS: Relation Importance. For each class the percentage of nodes for which a relation was defined important through either a forward approach or backward approach is reported. Only if the relation was found to be important for more than 50 % of nodes for the given class the percentage is reported.

Label	dimension	exhibition	maker	productionDateEnd	productionDateStart	title	object	proxyFor	proxyIn	type
Decorative art			0.88							
Prints										
Historical Artifacts				0.6						
Drawings							0.84		1	1
Metallic Art						1				
Books and Documents	0.96	0.44								
Paintings										
Photographs					0.76			1		

Fig. 11. MDGENRE comparison of relation distribution for node for the movie *Around the World in 80 Days*



5 Conclusion and Discussion

In this section, the conclusions derived from the experimental results will be presented. Subsequently, the limitations of this work will be introduced, together with a discussion on possible future work directions.

5.1 Answering the Research Questions

This study introduced two new methods to tackle important challenges in the realm of relational graph neural networks and their interpretability. These methods were developed to provide answers to the Research Questions outlined in

Section 1. This involved the implementation of RGCNExplainer and its utilization to explore how injecting knowledge impacts RGCN explanations (RQ 1). Additionally, the study explored the relations that define a class by conducting a statistical analysis of RGCN explanations and implementing the Relation Attribution method (RQ 2).

Firstly, the RGCNExplainer has been proposed as an approach to generate explanations for the RGCN model's node classification task. This method tackles the issue of identifying the most influential edges that contribute to the model's predictions for a given node.

Additionally, an exploration into the incorporation of prior knowledge into RGCN explanations has yielded interesting insights. While integrating prior knowledge about the input graph through mask initialization did not consistently improve explanations, promising results were observed when injecting specific domain knowledge by assigning higher weights to edges of certain relations in the mask.

The second part of this research delved into identifying crucial relations for the RGCN model within distinct classes. The analysis of relation distribution across explanation subgraphs yielded insightful results, which were - in most of the cases - confirmed by the results of the Relation attribution approach. It was found certain relations have a pivotal role in the RGCN model's classification process.

In summary, this study contributes to the advancement of interpretable relational graph neural networks through the introduction of the RGCNExplainer and the exploration of the Relation Attribution method. The insights gained from these methods deepen our understanding of model behavior and pave the way for further progress in the field of explainable AI and graph-based machine learning.

5.2 Limitations and Future Work

While the presented methods have demonstrated promising results in enhancing the interpretability of relational graph neural networks, certain limitations and avenues for future exploration should be acknowledged.

1. Computational Cost: One notable limitation is the computational cost associated with generating explanations using the RGCNExplainer. The process of identifying influential edges and generating explanation subgraphs can be resource-intensive, especially for large-scale graphs. This may lead to challenges in real-time or resource-constrained applications. Future research could explore optimizations to mitigate this computational burden and improve the efficiency of the explanation generation process.
2. Dataset-Specific Challenges: It is important to highlight that, while the proposed methods exhibited success on various datasets, there were instances where the RGCNExplainer did not yield satisfactory results. This could be attributed to suboptimal hyperparameters or specific characteristics of the dataset. Addressing this challenge involves navigating the impracticality of

hyperparameter tuning for each node to be explained, as this approach can be computationally prohibitive.

3. Hyperparameter Optimization: As hyperparameter tuning requires training a separate model for each node to be explained, it presents a significant practical challenge. The need to optimize hyperparameters for each individual node can be cumbersome and resource-intensive, making it impractical for large-scale applications. Future work could investigate techniques to automate or streamline hyperparameter optimization processes, potentially leveraging transfer learning or meta-learning strategies.
4. Robustness and Generalizability: Another avenue for future exploration is enhancing the robustness and generalizability of the proposed methods across diverse datasets and problem domains. Ensuring that the methods perform well consistently across various scenarios and data distributions is crucial for their real-world applicability.
5. User-Centric Explanations: Additionally, an exciting direction for future research involves tailoring explanations to suit the needs and preferences of end-users. Developing methods to customize explanations based on user feedback or domain-specific requirements could further enhance the usability and effectiveness of the interpretability techniques.

In conclusion, while this study has proposed methods to enhance the interpretability of relational graph neural networks, there remain challenges to address and opportunities for future research. By tackling issues related to computational cost, dataset-specific challenges, hyperparameter optimization, robustness, and user-centric explanations, RGCNExplainer can be enhanced in order to allow for the RGCN model to become more transparent and comprehensible.

References

1. "Triply." [Online]. Available: https://triplydb.com
2. T. Kipf, *Deep learning with graph-structured representations*, 2020.
3. D. Fensel, U. Şimşek, K. Angele, E. Huaman, E. Kärle, O. Panasiuk, I. Toma, J. Umbrich, and A. Wahler, *Introduction: What Is a Knowledge Graph?* Cham: Springer International Publishing, 2020, p. 1–10. [Online]. Available: https://doi.org/10.1007/978-3-030-37439-6_1
4. A. Gupta, P. Matta, and B. Pant, "Graph neural network: Current state of art, challenges and applications," *Materials Today: Proceedings*, vol. 46, p. 10927–10932, Jan 2021.
5. M. Schlichtkrull, T. N. Kipf, P. Bloem, R. v. d. Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," no. arXiv:1703.06103, Oct 2017, arXiv:1703.06103 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1703.06103
6. S. Bloehdorn and Y. Sure, "Kernel methods for mining instance data in ontologies," in *The Semantic Web*, K. Aberer, K.-S. Choi, N. Noy, D. Allemang, K.-I. Lee, L. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, p. 58–71.
7. P. with code. (2023) Node classification on aifb leaderboard. [Online]. Available: https://paperswithcode.com/sota/node-classification-on-aifb
8. H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," no. arXiv:2012.15445, Jul 2022, arXiv:2012.15445 [cs]. [Online]. Available: http://arxiv.org/abs/2012.15445
9. I. Tiddi and S. Schlobach, "Knowledge graphs as tools for explainable machine learning: A survey," *Artificial Intelligence*, vol. 302, p. 103627, Jan 2022.
10. R. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," no. arXiv:1903.03894, Nov 2019, arXiv:1903.03894 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1903.03894
11. T. Thanapalasingam, L. v. Berkel, P. Bloem, and P. Groth, "Relational graph convolutional networks: a closer look," *PeerJ Computer Science*, vol. 8, p. e1073, Nov 2022.
12. D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledgebase," *Commun. ACM*, vol. 57, no. 10, p. 78–85, sep 2014. [Online]. Available: https://doi.org/10.1145/2629489
13. "Dbpedia." [Online]. Available: https://www.dbpedia.org
14. T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," no. arXiv:1609.02907, Feb 2017, arXiv:1609.02907 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1609.02907
15. D. X. W. Y. B. Z. H. C. D. Luo, W. Cheng and X. Zhang, "Parameterized explainer for graph neural network," *Advances in Neural Information Processing Systems*.
16. N. D. C. M. S. Schlichtkrull and I. Titov, "Interpreting graph neural networks for nlp with differentiable edge masking," *International Conference on Learning Representations*, 2021.
17. M. K. T. Funke and A. Anand, "Hard masking for explaining graph neural networks," 2021. [Online]. Available: https://openreview.net/forum?id=uDN8pRAdsoC

18. A. Z. X. H. X. Wang, Y. Wu and T. seng Chua, "Causal screening to interpret graph neural networks,," 2021. [Online]. Available: https://openreview.net/forum?id=nzKv5vxZfge
19. J. W. K. L. H. Yuan, H. Yu and S. Ji, "On explainability of graph neural networks via subgraph explorations," *Proceedings of The 38th International Conference on Machine Learning*, 2021.
20. A. Seeliger, M. Pfaff, and H. Krcmar, *Semantic Web Technologies for Explainable Machine Learning Models: A Literature Review*, Oct 2019.
21. P. Wang, Q. Wu, C. Shen, A. Dick, and A. v. d. Hengel, "Explicit knowledge-based reasoning for visual question answering," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, p. 1290–1296. [Online]. Available: https://doi.org/10.24963/ijcai.2017/179
22. N. Díaz-Rodríguez, A. Lamas, J. Sanchez, G. Franchi, I. Donadello, S. Tabik, D. Filliat, P. Cruz, R. Montes, and F. Herrera, "Explainable neural-symbolic learning (x-nesyl) methodology to fuse deep learning representations with expert knowledge graphs: The monumai cultural heritage use case," *Information Fusion*, vol. 79, p. 58–83, Mar 2022.
23. P. Bloem, X. Wilcke, L. van Berkel, and V. de Boer, *kgbench: A Collection of Knowledge Graph Datasets for Evaluating Relational and Multimodal Machine Learning*, ser. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, vol. 12731, p. 614–630. [Online]. Available: https://link.springer.com/10.1007/978-3-030-77385-4_37
24. A. Jović, K. Brkić, and N. Bogunović, "A review of feature selection methods with applications," in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2015, pp. 1200–1205.
25. G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers Electrical Engineering*, vol. 40, no. 1, p. 16–28, 2014.

A Acknowledgements

I would like to express my gratitude to all those who supported and guided me throughout the process of this study. My sincere thanks go to Wouter Beek and Kathrin Dentler, along with the entire team at Triply, for their insightful teachings on Linked Data and unwavering support. I am deeply thankful to Ilaria Tiddi, who extended her support and constructive feedback even during her maternity leave and vacations. I want to give a special thank you to Taraneh Younesian for helping me with the technical aspects of this work and for her consistent feedback and support. Lastly, I'm thankful to my family and friends for their ongoing encouragement and support throughout this journey.