UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

Explaining Predictions from Node Classification with Knowledge Graph Embeddings

Filipe José Bastos Benvindo Paulino

Mestrado em Ciência de Dados

Dissertação orientada por:
Prof.ª Doutora Cátia Luísa Santana Calisto Pesquita

2024

Acknowledgments

First of all, I want to thank my supervisor, Prof. Cátia Pesquita, for the trust and for the opportunity to participate in this challenging project, and for the outstanding guidance and inspiration in every step of the way.

I also want to thank the LiSeDa group for their camaraderie and inspiration. Especially to Rita Sousa, co-author for the research paper that this work originated, for her constant support and solidarity.

I would like to acknowledge the LASIGE unit and community, for all the welcoming experiences, resources and support. I'm proud to be a part of this group.

I also acknowledge all the exceptional professors of the Master Programme in Data Science. And express my appreciation to fellow students that contributed to my success.

I am grateful for the funding provided by the KATY project which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017453.

Finally, my deepest gratitude to my family and friends, for their unconditional support and unwavering belief in me throughout this endeavour.

Abstract

Current AI solutions, especially black-box models, show a set of limitations that may include biases and lack of transparency. Especially in high-risk applications, such as medical diagnosis or infrastructure management, the need to understand the reasoning of such models gives rise to post-hoc explanation methods. Knowledge Graphs represent real-world entities and their relations in a graph structure with explicit semantics and support several applications from recommendation systems to question-answering and data mining. Knowledge Graph Embeddings have emerged as a solution for representing entities for downstream tasks, such as link prediction or node classification, by producing vector representations of entities in an embedding space that aims to conserve syntactic and structural properties while being amenable to further computation. However, they do so at the cost of the inherent explainability of knowledge graphs: most approaches result in meaningless vectors.

This work builds on current approaches in order to explain node classification prediction models based on Knowledge Graph Embeddings, a problem not tackled by current methods. It applies Explainable AI paradigms to design, implement and test two novel methods, LOFI, and C-KEE, that are perturbation-based and that use counterfactuals, in the form of necessary and sufficient explanations, to identify the set of facts or entities present in the Knowledge Graph that more profoundly affect an entity's classification.

Extensive experiments were conducted using different benchmark datasets. In the end, LoFI showed mixed results, being able to successfully generate sufficient explanations only. With C-KEE, a superior solution was achieved, given that it significantly outperformed baselines in almost all scenarios, including several representative Knowledge Graph embedding methods, while being able to produce empirically sound and intuitive explanations in minimal computational processing time.

Keywords: knowledge graph, knowledge graph embeddings, graph neural networks, machine learning, explainable artificial intelligence

Resumo Alargado

Este trabalho surge no contexto dos recentes avanços em Inteligência Artificial nas mais diversas áreas do conhecimento e tecnologia. Um exemplo motivador deste trabalho são as aplicações de elevado risco, tais como diagnóstico médico ou gestão de infraestruturas em que a Inteligência Artificial pode ser usada, por exemplo, em sistemas de recomendação, de pergunta-resposta ou de automação. Acontece que a maioria dos atuais algoritmos e soluções de elevado desempenho em Inteligência Artificial podem carecer de transparência, já que o seu desenho veda aos programadores ou utilizadores a compreensão concreta dos mecanismos ou causas específicos por detrás de um determinado resultado, como por exemplo uma sugestão de terapia para um determinado paciente. Esta limitação é uma entre várias existentes no atual paradigma da Inteligência Artificial, das quais também se destaca a hipótese de propagação de enviesamentos e preconceitos pessoais que podem existir nos dados de treino dos modelos.

As redes de conhecimento representam entidades do mundo real e as suas relações numa estrutura de grafo com uma semântica formal e explícita. Sendo altamente interpretáveis, estas redes são tomadas como uma solução para desenvolver Inteligência Artificial cujo funcionamento é compreensível por humanos. Este trabalho surge no contexto da classificação de entidades usando soluções que consistem em modelos de vetorização de redes de conhecimento acoplados com modelos preditivos de aprendizagem automática. Estes modelos de classificação são um exemplo onde a otimização do desempenho de soluções conduz à utilização de algoritmos opacos que resultam na perda da interpretabilidade original. A vetorização de redes de conhecimento é uma solução para representar as entidades presentes nas redes num formato propício para uso em tarefas subsequentes, tais como previsão de relações ou classificação de entidades, e que procura conservar as propriedades sintáticas e semânticas da rede. No entanto, mesmo que o algoritmo subsequente responsável pela tarefa de previsão seja interpretável, qualquer hipótese de transparência é perdida no processo de geração dos vetores e na representação sub-simbólica dos próprios vetores. O objetivo de aliar o uso das soluções correntes com a compreensão da lógica implícita em tais soluções gera a necessidade de criação de métodos de inteligência artificial explicável, que procuram gerar explicações para modelos opacos existentes (explicações post-hoc).

Existem poucos métodos que tentem implementar explicações post-hoc para soluções que usam vetores de redes de conhecimento. As ferramentas mais populares estão desenhadas para aplicações que usam dados como texto, imagens ou tabelas, mas devido ao tipo de formato fundamentalmente diferente das redes de conhecimento as ferramentas atuais não são trivialmente adaptáveis a estas. Outro dos principais desafios é que estes modelos de vetorização das redes de conhecimento são, em muitos casos, transdutivos e portanto perturbações nos dados de entrada implicam o retreino completo do modelo, o que se traduz num processo demasiado lento e impraticável quando associado às abordagens mais usualmente empregues por métodos de explicabilidade. Por outro lado, a literatura está muito mais orientada

para a tarefa de previsão de relações, e nenhum dos métodos existentes se foca ou admite a geração de explicações para a tarefa de classificação de entidades. Uma das dificuldades adicionais na explicação desta tarefa é o facto de usar dois modelos distintos que devem ser explicados integradamente, já que as características naturalmente interpretáveis existem à entrada do primeiro modelo mas a possibilidade de avaliação das perturbações tem de ser aferida à saída do segundo modelo.

O objetivo deste trabalho é desenvolver explicações para previsões individuais, tendo em conta contextos como sistemas de recomendação, apresentando um método que seja relativamente independente do tipo de modelos de vetorização de redes de conhecimento e de modelos de aprendizagem automática usados nas previsões. O modelo proposto tem como preocupação gerar explicações em tempo útil tendo em conta as necessidades típicas de uma aplicação real, e gerar explicações que promovam uma compreensão fácil e intuitiva da previsão em questão.

Uma das abordagem mais comuns em explicabilidade é o uso de perturbações sobre o modelo de previsão, e que genericamente consiste em investigar como pequenas mudanças nos dados de entrada para o modelo de previsão afetam os seus resultados. A metodologia proposta neste trabalho consiste na aplicação desta abordagem para desenvolver um modelo explicativo das previsões. As explicações são compostas por exemplos que podem ser factos ou entidades presentes na rede de conhecimento inicial. As explicações usam métodos contrafactuais com explicações necessárias ou suficientes, que respondem às questões sobre *"o que aconteceria à classificação da entidade se esta não fosse definida por estes factos ou entidades"* ou *"o que aconteceria à classificação da entidade se esta fosse apenas definida por estes factos ou entidades"*.

Especificamente no primeiro método proposto, LoFI, as perturbações são factos na vizinhança direta da entidade a explicar. Este método consiste em aproximar perturbações na rede de conhecimento original, por exemplo, retirando um dos factos e medindo o impacto dessa alteração na resposta do modelo de previsão. LoFI explica soluções baseadas em caminhos no grafo, onde as perturbações são materializadas com um retreino parcial apenas dos vetores afetados pelo facto em questão, usando métodos de aprendizagem em linha que permitem contornar o problema do retreino completo dos vetores da rede de conhecimento. No segundo método proposto, C-KEE, as perturbações são as entidades na vizinhança direta da entidade a explicar. Este método substitui a representação vetorial original da entidade a explicar por uma representação vetorial que agrega as representações das entidades na sua vizinhança direta. O método C-KEE, tal como proposto, é capaz de gerar explicações para qualquer combinação de vetores de redes de conhecimento e modelo de aprendizagem automática, sendo as perturbações aplicadas sobre a representação global, evitando assim completamente a necessidade de retreino. Ambos os métodos permitem obter explicações com mais de um facto ou entidade, mas o comprimento máximo das explicações é mantido abaixo de um limiar que promove uma fácil interpretação da explicação.

Numa primeira fase, ambos os modelos são avaliados em comparação com explicações obtidas aleatoriamente e em comparação com soluções alternativas que permitem explicabilidade, mais concretamente, redes neuronais para grafos usando métodos de explicabilidade já estabelecidos. Nesta primeira fase, LoFI demonstrou resultados parcialmente positivos, enquanto que C-KEE foi superior em praticamente todos os casos de avaliação.

Numa segunda fase, o método que se revelou mais promissor, C-KEE, é sujeito a uma exaustiva avaliação que permite demonstrar a sua generalização para combinações de vários métodos de vetorização

de redes de conhecimento e de aprendizagem automática. Adicionalmente, nesta fase também são explorados o impacto de diferentes comprimentos das explicações e o impacto de diferentes definições do sucesso de uma explicação. Observa-se, nesta fase, que com o C-KEE é possível obter um desempenho significativamente melhor que as alternativas em praticamente todos os cenários, produzindo explicações intuitivas e empiricamente sustentadas, usando um tempo de processamento mínimo.

A contribuição deste trabalho é o desenho, implementação e validação de dois novos métodos de explicabilidade, LoFI e C-KEE, capazes de explicar soluções típicas para classificação de entidades baseada em vetores de redes de conhecimento, investigando diferentes tipos de explicações contrafactuais, e onde foi possível produzir explicações eficientes e orientadas para uma interpretação intuitiva das decisões dos modelos. Destaca-se, com o C-KEE, o desenvolvimento e validação de uma nova abordagem que engloba uma representação alternativa das entidades da rede, que potencia a geração de explicações de forma integrada, enquanto ao mesmo tempo evidencia um desempenho comparável ou superior na tarefa original de classificação.

Trabalho adicional poderá ser feito na avaliação dos métodos desenvolvidos, através de estudos com utilizadores, como forma de validação adicional e também com o intuito de obter um conjunto de opiniões imparciais que possam conduzir à melhoria dos métodos propostos.

Este trabalho produziu um artigo científico, o qual foi submetido e está em processo de revisão por pares.

Palavras-chave: redes de conhecimento, vectores de redes de conhecimento, redes neuronais para grafos, aprendizagem automática, inteligência artificial explicável

Contents

List of Figures

List of Tables

Acronyms

AI Artificial Intelligence.

GCN Graph Convolutional Network.

GNN Graph Neural Network.

KG Knowledge Graph.

KGE Knowledge Graph Embeddings.

LP Link Prediction.

ML Machine Learning.

NC Node Classification.

NN Neural Networks.

RDF Resource Description Framework.

SL Supervised Learning.

XAI Explainable Artificial Intelligence.

Chapter 1

Introduction

1.1 Context and Motivation

This work arises in the context of the advancements of Artificial Intelligence (AI) in every technological area [65]. As the range of applications increases and new problems in different areas are tackled, so do new challenges arise [65]. The use of AI in high-risk applications like healthcare [9, 39] or infrastructure [46], is one of those challenging areas where, in several situations, AI is to be used as a decision support tool, for example with recommendations targeted at a human specialist.

However, AI has an important set of limitations [40, 72] that hinder its adoption and limit its practical value. Some of these are that it may fail due to failed assumptions, algorithmic errors or statistical inaccuracies [40], it may exacerbate people bias for example by reinforcing existing bias in the data and encapsulate it under the guise of a scientific tool, it may lack transparency [70] in that it may be difficult to understand its logic, and it may trigger a lack of trust from the users when they are not able to understand its inner workings or when it produces a failed prediction, among other issues. In areas like decision support, real-world applications are seldom possible without addressing these limitations. In such cases, for the AI to have practical value, the human needs to understand the reasoning behind the AI recommendation, which requires from the AI a logic that is transparent and understandable [2].

It so happens that many of the best performing AI solutions is based on Machine Learning (ML) models that are black-box, meaning that their logic is too complex to be understood by humans. For example, Neural Networks (NN) [35] are black box and a current major industry trend. In some situations, even more than one black box algorithm may be combined to extract knowledge and produce the kind of valuable predictive capabilities that are so sought after when using these tools. In the context of those high-risk applications, should the user prefer worse performing but interpretable models or can the usability of these complex models be enhanced? It is here that Explainable AI (XAI) [17, 48] comes into context, to develop explainability methods that enable the use of these opaque models in critical decisions by providing an understanding of what the model is doing.

One of the current trends in AI are Knowledge Graphs (KGs) [67, 59]. KGs use a graph structure to organize collections of real world entities and relations between those entities, according to a formal definition of concepts called ontology [25]. Being highly interpretable and having a rich semantics, KGs help build AI that can be understandable by humans [67], but while trying to maximize performance of the AI using KGs, black box methods are frequently used and the interpretability is lost [53, 21].

ML black-box models used in conjuntion with KGs are mainly of two types [25]: Knowledge Graph Embeddings (KGE) [74] or Graph Neural Networks (GNNs) [78]. KGE models transform a Knowledge Graph into dense vector representations called embeddings, for example allowing the downstream use of another ML algorithm capable of Supervised Learning (SL) to make predictions based on those embeddings, but even if the SL model is interpretable, any chances of transparency are lost when applying the KGE model to generate the embeddings, so these solutions are performant but opaque. GNNs are Neural Networks designed to take KGs directly as input data and output a prediction. Both of these solutions are recent when compared to ML for text or image, and so are the XAI methods that try to explain the outcomes of these solutions.

XAI is a recent but fast growing field. Concretely, it's about deriving explanations for existing models that are black box such as KGEs or GNNs. These types of explanations are called post-hoc explanations. Many of the major approaches used to extract explanations from AI models use attribution methods (e.g. [34, 43, 51, 80, 83]). These methods investigate how small changes in some model input data impact that model output results. Many of these methods are perturbation-based meaning that they rely on the analysis of how small perturbations in the input data impact the outcome. Popular tools include general-purpose methods such as LIME [51] or SHAP [34] and a large number of work/methods that are model specific, with many solutions especially in Neural Networks [28]. The bulk of these works develop techniques to explain models that consume text or image data, but are also easily adaptable for tabular data. Compared to these, graph data has a more complex topology, it is not sequential like text, or "grid-ordered" like image or tabular, and has very different types of representations like homogeneous graphs or RDF graphs. Because of this, generic explainability methods usually are not trivially adaptable to graphs.

In this regard, most work has been devoted to explaining GNNs, for example GraphLIME [27] which is an adaptation of LIME [51] able to produce non-linear explanations, a system that is more suitable for graphs; GNNExplainer [80] which combines node feature and edge explanations and favours explanations as collections of edges and was the first explainer designed from scratch to explain GNN outputs; SubgraphX [83] which targets true subgraph explanations, not just collections of edges. It's worth noting that although GNNs can also be used to inductively generate embeddings in an unsupervised manner, the existing explainers all focus on explaining the results of end-to-end prediction tasks.

KGE are commonly used in tasks such as Link Prediction (LP) or Node Classification (NC). Methods trying to implement a post-hoc explanation of AI models that rely on KGE are very limited. One of the main challenges in explaining such models is that KGE generation is a transductive task, so any changes to the input data requires the full retraining of the embeddings which is computationally very expensive. ExplaiNE [30] is an explanation method designed for Link Prediction tasks based on graph embeddings and, according to the conducted literature review, the first of its kind. Kang and Park [31] propose another solution for graph embeddings, uniquely interesting in that it tries to explain the embeddings algorithm itself, not tied to any prediction task or application. Kelpie [55] is an explanation framework for Link Prediction tasks on KGE, that focuses on trying to reduce the search space for explanations and limit the retraining of the embeddings as optimization measures. One of the concerns common to all these works is to optimize the retraining by implementing only a partial retraining of embeddings. Some of the highlighted works are concerned with graph embeddings and do not generalize to KGE due to the more

complex nature of KGE in terms of structure. Graph embeddings can be represented by simple adjacency matrices while KGE have notably elaborate representations such as RDF graphs [1]. Remarkably, none of the solutions found in the literature for post-hoc explanations are designed to generate explanations for Node Classification tasks that use KGE. Node Classification tasks require an additional Supervised Learning model and use targets that are not part of the KGE which changes the nature of the used predictive models, for which none of the presented explainability methods are adequate.

1.2 Problem Definition

Considering a KG as a labelled directed graph $KG = (E, R, F)$ where E is the set of nodes representing entities, R is the set of relations, and F is the set of facts (edges) connecting nodes through relations. A fact in F is denoted by a triple of the form $< h, r, t >$ where h is the *head* and t is the *tail* with $h, t \in E$, and r is the relation with $r \in R$.

A KGE is a representation of a KG entity or relation in a dense (continuous) numeric vector (embedding) e with dimension d, obtained with a KGE model.

Node classification uses KGEs as input features to a supervised learning model that will learn to classify an entity according to a property external to the KG. In this regard, an entity e in E, belonging to the KG, is associated with a class c in C. In step 1, the KGE model encodes e in a sub-symbolic representation e_{sub}. In step 2, the SL model learns e_{sub}, and as a consequence e, as belonging to class c.

So, because the target of node classification is not part of the KG and because KGE models are coupled with classifiers in node classification tasks, any perturbations introduced should be directed towards the KG itself. However, the evaluation of the perturbation's impact must be conducted in relation to the output generated by the supervised classifier. This interplay between KGEs and ML models introduces an additional layer of complexity that is not solved in the current approaches.

1.3 Objectives and Contributions

Taking into account the limitations in the literature, namely the lack of explanation models for KGE used in Node Classification tasks, the objective of this work is to develop an original explainability method designed to provide insight about the inner workings of AI models based on KGE when applied to Node Classification using downstream SL models.

Local explanations are concerned with explaining the prediction for some particular instance, as opposed to explaining the general logic behind a model's behaviour. Local explanations are more valued in several explainability contexts like for example recommendation systems. Also, most explainability solutions in the literature are usually concerned with this type of explanation. For these reasons this work focuses on **local explanations**.

It should be **model-agnostic**, ideally for a wider range of applications, because the capability for explanation should not limit the range of KGE or ML models used to tackle some problem and particularly in Knowledge Graph-based Machine Learning there are at least as many models as there are different types of KGs.

[1] `https://www.w3.org/RDF/`

The major challenges are the ability to produce relevant explanations faithful to the model, and able to produce individual explanations in a useful time frame for the purpose of a practical application like for example a recommendation system.

The research question that guides this endeavour is: can we generalize current model agnostic, post-hoc, perturbation based, local explanation, XAI approaches to KGE based Machine Learning for Node Classification?

The major contribution of this work is the design, implementation and validation of two novel methods for explainability, designated Local Facts for Interpretability in Node Classification with KGE (LoFI) and Classification with Knowledge Graph Embeddings Explained (C-KEE). The proposed methods are able to explain common solutions that use Supervised Learning models and transductive embeddings obtained from KGE models and used for tasks of Node Classification. LoFI explains solutions based on walk-based KGE but should generalize to other types of KGE solutions with minimal adaptations. C-KEE, as is currently proposed, is truly model-agnostic and should be able to explain any KGE+NC solution.

1.4 Dissertation Outline

This document is organized in the following order:

- Chapter 1 - Introduction of the work, including the context and main objectives behind the work;
- Chapter 2 – Background knowledge in the form of base concepts and principles required to understand the remaining of the document, such as Knowledge Graphs, KGE and Explainable AI;
- Chapter 3 – Literature review of the critical works on which this work builds upon, mainly focused on explainability solutions developed in recent years in the field of AI;
- Chapter 4 - Methodology used in the course of this work to develop, implement and validate the Explainable AI solutions;
- Chapter 5 - Results and discussion summarizing the most important findings during the validation step of the Explainable AI solutions; and
- Chapter 6 - Conclusion of the work done with a summary of the main accomplishments and plans to build on top of the current work.
- Appendix A - Additional information on parameter optimization of the KGE models
- Appendix B - Accuracy results of the LoFI and C-KEE explainer models
- Appendix C - Algorithm for the sufficient explanations generated with C-KEE
- Appendix D - Additional information on the training parameters for the additional results of the C-KEE explainer model
- Appendix E - Additional results of the C-KEE explainer model
- Appendix F - Research paper based on this work and submitted to ACM KDD 2024

Chapter 2

Background

This chapter presents a brief introduction to the different theoretical and practical concepts or solutions in ML and AI that were explored with this work and that provide a foundation for it. Section 2.1 introduces Knowledge Graphs. Section 2.2 briefly introduces Knowledge Graph Embeddings and Graph Neural Networks, the most popular ML methods that rely on KGs. Finally, Section 2.3 addresses some important ideas and concepts in XAI that are relevant for the work.

2.1 Knowledge Graphs

2.1.1 Introduction

Knowledge Graphs (KG) are a hot topic in AI research and applications. There are divergent opinions as to what defines a KG and how they differ from previous concepts in knowledge and graph databases [15, 25]. This work considers a general definition proposed by Hogan *et al.* [25] stating that a Knowledge Graph is "a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities". A more formal definition can be as follows:

Definition 1 - Knowledge Graph.

A Knowledge Graph $G = (E, R, F)$ is a directed graph with E the set of *entities* and R the set of *relations*. $F \subseteq VxExV$ is the set of edges, also called *facts*, that link entities through relations.

A fact is denoted by a triplet of the form $< h, r, t >$, where h is the *head* and t is the *tail* with $h, t \in E$, and r is the relation with $r \in R$. The *data* represented in the Knowledge Graph are the instances of knowledge composed of entities and relations. An example would be the entities *Lisbon* and *Portugal* connected by a relation *is_capital_of*. On top of these instances of data, the Knowledge Graph can express more general knowledge using *ontologies* or *rules*. An example would be the concepts *capital* and *city* connected by a meta-relation *is_a*. Finally, in the KG, entities data is mapped to concepts in the ontology. One of the main features of Knowledge Graphs is thus the combination of meta-relations between concepts in the form of ontologies with data in the form of entities mapped to those concepts. A visual example of such KG is shown in Figure 2.1.

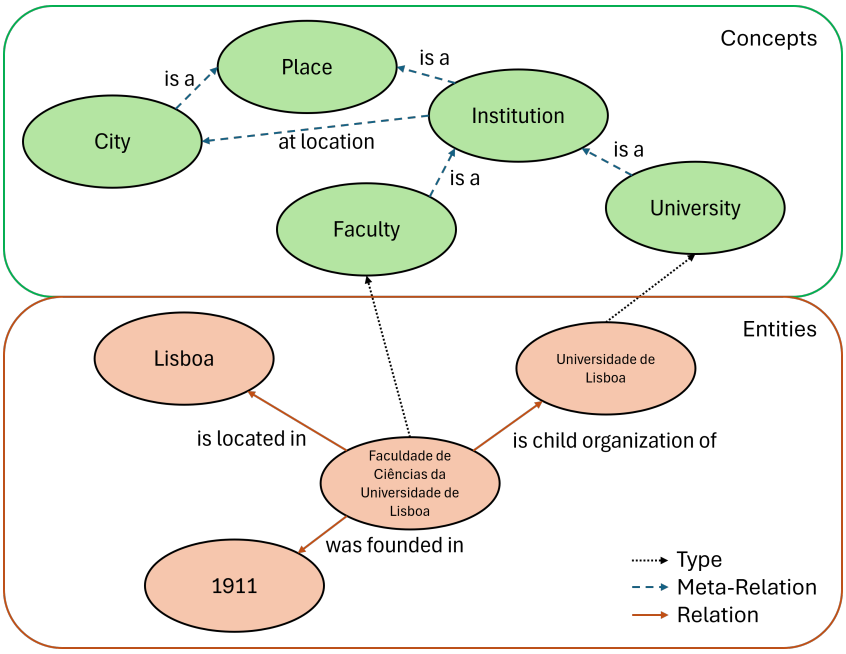

Figure 2.1: Example of a Knowledge Graph with concepts and entities.

Contrary to other types of networks, in the vast majority of KGs, and including all the largest KGs and industry benchmark KGs, the facts or edges in the KG do not have weights associated to them. A fact either exists or not and this is strongly tied to the concept of facts as instances of real world knowledge, for example either Lisbon is the capital of Portugal or it's not, there are no uncertainty considerations.

Training by example is today one of the fundamentals of AI, and KGs combine the ML data-centric approach with the rich semantics of the ontologies and the principles of Semantic AI thus creating a very powerful solution for solving tasks such as recommendation systems or predictive modeling [25].

2.1.2 Types of KGs

The Resource Description Framework (RDF) graph and the Property Graph (PG) are the most common data models used to represent Knowledge Graphs. The RDF graph is a type of directed edge-labelled graph where each element of the triple is uniquely identified by a Uniform Resource Identifier (URI). RDF graphs focus on standardization to ease the interchange, merging and sharing of KG data and are the reference model for Knowledge Graphs built on the principles of the Semantic Web.

Property graphs are a more flexible graph alternative whose major focus is the possibility of having property value pairs included in any node or edge. Property graphs do not have standards, including no defined formal semantics, meaning that each vendor solution is defined by its own rules and do not adhere to a standard so it cannot be integrated with other sources.

There is an important trend for integration of multiple data sources [49, 22] and the RDF graph, being built upon Semantic Web Technologies, has advantages over the other types of Knowledge Graphs such as standardization and semantic interoperability [12]. For this reason the use of RDF KGs is preferred in this work.

2.1.3 Main Applications of KGs

KGs have many practical uses divided in two major categories: deductive knowledge and inductive knowledge [25]. Deductive knowledge broadly concerns the use of logical frameworks such as ontologies in order to deduce new facts (entailments) from existing Knowledge Graphs. This is generally accomplished by reasoning directly over the Knowledge Graphs using rules or logic. Inductive knowledge concerns the use of various methods able to take a set of input observations and make predictions based on some observed pattern. Recently, in this category, the topic of ML methods especially designed for graphs and Knowledge Graphs have been the target of much attention and research. This work is also placed under this topic.

Applications using Knowledge Graphs include Link Prediction, Entity or Node Classification, Triple Classification and Entity Resolution [74]. Link Prediction seeks to find missing facts in the Knowledge Graph, it is usually the task of taking some head h or tail t that is most plausible for the incomplete triple $< ?, r, t >$ or $< h, r, ? >$, respectively.

According to some authors, Node Classification is a subset of Link Prediction. As stated in Section 1.2, this work does not assume this to be true because it considers that Node Classification happens in the context of the ML task, not as a KG completion problem. As in typical ML problems, Node Classification is thus to predict the label of some node from within a subset of mutually exclusive classes (binary or multi-class). In applications where classes can exist in conjunction the prediction task becomes a multi-label problem. Finally, Node Classification can also combine several of the previous tasks in multi-output classification problems.

2.2 Knowledge Graph-Based Machine Learning

Presently, there are two main approaches when using Knowledge Graphs in conjunction with Machine Learning. One is to use Knowledge Graph Embeddings, the other is to use Graph Neural Networks [25].

2.2.1 Knowledge Graph Embeddings

Knowledge Graph Embeddings are a technique that transforms the original graph representation into a dense numeric representation in the form of a set of vectors. Such techniques project a graph into a low-dimensional Euclidean space while attempting to preserve the inherent relational properties of the graph [8]. These are one of the most successful uses of Knowledge Graphs because these embeddings can be used as input to downstream tasks that rely on traditional Machine Learning. One important downside of using this dense representation is that it is sub-symbolic, so the meaning of the original graph is lost.

There exists a panoply of KGE algorithms that stem from different embedding techniques such as translational, semantic matching, deep learning and path-based [25, 74]. Translational models consider that *head* nodes are transformed into *tail* nodes by the *relation* that connects them [5]. Semantic matching models explore semantic similarity measures to match entities using a matrix of relations that captures the interactions between the entities [74], deep learning models, like SDNE [73] use neural networks to try to improve the capture of highly non-linear network structure when compared to shallow methods, finally path-based methods use random walks to capture the structure of the KG, like in RDF2Vec [53].

These methods are also known for their transductive learning nature, in that the model performs a reasoning from one instance to another, by nature considering that all instances are available to reason with, and as a consequence its learning procedure by default does not generalize to new, unseen instances. What this means is that these methods require that all the instances of interest to be present when learning the embeddings and as a consequence, to admit a new instance in the training set usually implies the need to train all the embeddings from scratch.

Graph Walks in KGE

This work is particularly interested in neural and language models that use graph walks to sample the KG, and which have proven useful in downstream classification tasks [61, 9], such as RDF2Vec [53].

One way of extracting the set of sequences for a graph is to use graph walks. For a vertex $v \in V$, given a depth d, several algorithms can be used to compose the set of sequences for v. The algorithm generates paths by exploring the edges, starting with v, and in the case of KGs, the edges considered are always outgoing edges, and for each edge there should be a connected vertex. Graph walks are also called random walks due to the unbiased manner with which the neighbours are selected using breadth-first or depth-first algorithms. This $< v, e, v >$ is similar to the $< h, r, t >$ triple previously mentioned. The depth d imposes the limit to the number of triples in each sequence.

Definition 2 - Graph Walks.

Taking a KG, each vertex $v \in V$ originates a set of sequences S_v, each sequence $s \in S_v$ being composed of paths originating from the vertex v.

The exploration of a graph is commonly limited by a total number of sequences S_v that is to be extracted for each v, for large graphs this is a key feature of the algorithm where the randomized approach is used to generate the limited set of sequences. The random walks are thus a collection of paths extracted from the graph that ultimately represent the graph as a set of sequences.

RDF2Vec-Based Embeddings

RDF2Vec [53] is based on the idea of extracting sub-structures from an RDF graph by converting the graph into a set of sequences of its entities and relations. The generation of the embeddings is composed of two major steps, somewhat independent of each other, the first is to materialize the key idea which is to map the KG into a set of sequences of its entities and relations, the second is to use the set of sequences to map the entities contained there into a continuous numerical vector representation. In RDF2Vec, the set of sequences is usually obtained using random walks, whereas the embedding learning step employs Word2Vec.

Word2Vec [36] is a model architecture that uses a simple neural network for computing continuous vector representation of words from large quantities of text. This architecture encodes each word into a sub-symbolic representation able to improve the accuracy of various syntactic and semantic word tasks. The model architecture of Word2Vec processes these large quantities of text using moving windows composed of the target word and its context (the words before and after). These windows function as

small sentences that are being fed to the neural network. For example, for a window of size 5, the word *t* can be predicted based on its context *t-2, t-1, t+1, t+2* in the CBOW architecture, or the context can be predicted from the word. For some path in a KG, as an example, this window can fit a sequence of two triplets < *Lisbon, is_capital_of, Portugal, part_of, Europe* > in which *Portugal* is the current word and the remaining entities and relations are its context.

In the context of RDF2Vec, the sequences obtained from the random walks are taken as natural language sentences and fed to Word2Vec [36]. This pipeline thus originates a sub-symbolic representation of each vertex or entity in the graph and the collection of these representations is therefore the sub-symbolic representation of the Knowledge Graph.

Node Classification with Knowledge Graph Embeddings

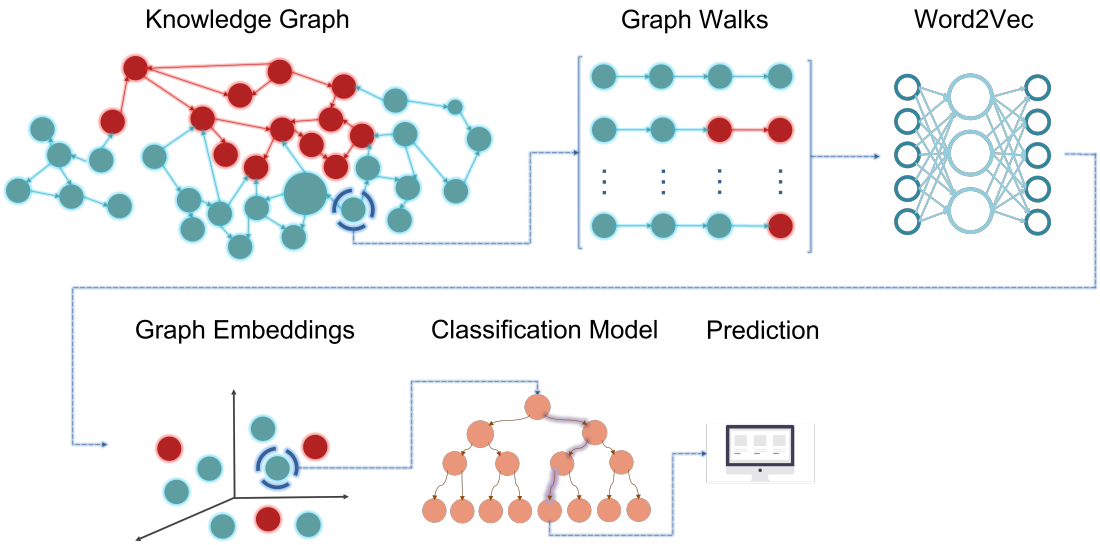

Figure 2.2: Overview of the pipeline for Node Classification with Knowledge Graph Embeddings generated using RDF2Vec.

Knowledge Graph Embeddings are easily used to perform a downstream Node Classification task that will rely on traditional classification ML models. Each embedding can be fed into the ML model as the set of features characterizing the entity associated with it. Node Classification using KGE is a two step process, first the embeddings model is applied as a Self-Supervised Learning task, and subsequently the classification model is learned and applied using the embeddings as inputs.

An overview of the process of generating a prediction from a KG using the combination of an RDF2Vec embeddings model and a downstream ML classification model is shown in Figure 2.2.

2.2.2 Graph Neural Networks

The alternative solution to the KGE and downstream SL model is to input the graph or Knowledge Graph directly into a Machine Learning algorithm and the main solution accomplishing this are Graph Neural Networks. One of the motivations behind this is to avoid the transductive nature of the KGE models. GNNs are generally inductive in that they naturally generalize for unseen nodes [21]. The general idea for GNNs is to aggregate the information of neighbouring nodes in order to generate a representation of the target node. The common steps used by these models are: 1. sample the neighborhood of the

target node; 2. aggregate its features using some specific aggregator function; and 3. predict the node label using the aggregated representation [21]. The Node Classification is directly embedded into the target node representation during training which means this approach could be said to synthesise the embedding and classification steps into one single model.

Graph Convolutional Network (GCN) [32] is a very popular type of GNN, that uses convolution operators and aggregation operators. And there are now many categories of GCNs, for example tailored to different types of graphs, including some solutions for Knowledge Graphs. A popular example of a GCNs specifically tailored to some types of Knowledge Graphs is R-GCN [57]. Another less known solution is the KGNN [33].

With this GNN approach, the understanding of what the AI is doing is lost in the Machine Learning model by using Neural Networks which are generally considered opaque models.

Considering these two main approaches it may be observed that both Knowledge Graph Embeddings and Graph Neural Networks lack interpretability and thus motivate the use of explainer models.

However, most GNNs have limitations on processing ontology rich graphs like the ones made possible with RDF graphs which is a major limitation in several applications. On the contrary, many KGE solutions easily support RDF graphs.

2.3 Explainable AI

XAI can be described in simple terms as the research field in Computer Science that is concerned with improving the human understanding of AI systems outputs and inner-workings [1].

Explainability has been present in AI since the beginning. In the early days Expert Systems explanations were intended to address questions such as the Why, What and How of Symbolic AI systems results [66]. With the rise of more powerful Machine Learning, explainability was traded for predictive power but more recently it has had a new boost in interest from the AI community, mainly repurposed as a way to understand those powerful but opaque ML systems [1].

From the AI limitations outlined, XAI major concern is to tackle the transparency and trust issues raised by the use of opaque models. Transparency can be understood as the degree of information that is given by a model about its inner workings [68]. Trust can be understood as the degree to which a user trusts the machine's output and to which it would follow its advice [24].

XAI methods are by definition a post-hoc tool. This is opposed to integrated methods which rely on models that explain themselves but that have fewer degrees of freedom and are thus incapable of generating more complex representations e.g. linear models, rule models [13]. XAI has vast amounts of methods but there are general approaches that help organize the field.

The scope of the explanation is divided in global vs local explainability. A global method is concerned with understanding the logic of the whole model and how it deals with any possible input [1]. While a local method is concerned with the model decision for a specific instance and the explanation only justifies that particular decision [1].

The methods can be be categorized according to their range of application as model-specific or model-agnostic. Model-agnostic means it ignores the actual model for which it is producing the explanation, making the explaining method general purpose. Model-specific means the explaining method

takes into account the type of black-box model that it's trying to explain and uses known specificities of that model's algorithms as foundational elements of the explaining method.

Perturbation and Attribution Methods

One of the most successful and widely used approaches to generate local explanations is to attribute resulting changes in the outcome of a model to some changes in the input. The systematic and intelligible connection (attribution) of output to input variables is the basis of attribution methods.

According to Jiménez-Luna *et al.* [29] there are three major attribution methods: gradient-based methods that compute the gradient of a network f in relation to the input x; surrogate methods that replace the original model with an interpretable model g; and perturbation-based methods that apply small modifications to the input and measure how those changes impact the output of the model. Simply removing or masking parts of the input is also a common technique in this approach. This is perhaps the most widely used approach to generate XAI methods.

In each of these methods, the goal is to find the most impactful input changes, often times in the form of an input feature or collection of features that are therefore deemed relevant to the model prediction being explained.

Chapter 3

Related Work

This chapter tackles the relevant literature review which for the context of this work is strongly tied with XAI research, publications and tools. In XAI, general purpose methods can be applied in a variety of domains, allow different types of input data and can explain a broad range of ML models. Specific methods are usually tailored for a particular problem and are not trivially adapted to other uses. To organize the most relevant literature for this work's context and purpose, the methods are divided according to the range of application, going from general-purpose presented in Section 3.1, to problem specific, namely XAI solutions for GNNs in Section 3.2 and XAI solutions for KGEs in Section 3.3. Finally, Section 3.4 addresses additional relevant topics in explainability and that are relevant for the course of this work.

3.1 General Purpose Explainable AI

Among the most general purpose methods the LIME [51] and SHAP [34] frameworks should be highlighted, as they are some of the most well-known contributions to XAI.

LIME [51] produces local and model-agnostic explanations. It is perturbation-based and uses a surrogate model that provides interpretability, it generates feature-based explanations. It allows for surrogate features and weighs samples importance according to distance metrics. Despite its generality, it needs to make algorithmic considerations according to the type of input data (tabular, text or image) it is trying to explain and for graph data it does not translate trivially because of the combination of features and graph topology. Also, this approach is grounded on the fact that the perturbed instances can be directly fed to the prediction stage of the models to explain, which is a very fast procedure, and that is why it uses a large amount of perturbations as a basis for the explanation model, but in transductive learning each perturbation implies the retraining of the model which would be prohibitive in regard to computational complexity.

SHAP [34] is based on the well known concept from game theory, Shapley values, and it is also local, model agnostic and perturbation-based. The explanation output is more meaningful than LIME and it does not rely on surrogate models making it more robust to explain non-linearity. But it makes use of more specific implementations for the different kinds of models being explained, and like LIME its approach is not directly applicable to graph data.

3.2 Explainable AI for Graph Neural Networks

The taxonomy of methods most relevant for generating GNN explanations is vast. According to Yuan *et al.* [82], for instance level explanations there are gradient-based (or backpropagation-based) methods [43], perturbation-based methods [80], decomposition methods [58] and surrogate methods [27]. For model level explanations there are generation methods [81]. Relevant works approximately corresponding with these approaches are highlighted in this section.

GraphLIME [27] is an explainer method that builds on LIME and its main focus is to provide non-linear explanations for graph models. But unlike LIME it uses a simple mask generator and no distance weighted samples. Its explanations are very simplistic focusing on a rank of node feature importance's. Most important, its explanations do not measure the importance of nodes or edges which especially for the case of KGs is seriously limited. This approach uses a surrogate model that is simple and interpretable to capture the most important features of the original model behaviour. By using a simple model it focus on explaining a delimited fraction of the original model complexity assuming that the targeted fraction is indeed simple enough to be captured by the explainer. It is also a perturbation-based technique.

GraphSVX [14] is a perturbation-based and surrogate model explainer that is able to efficiently compute Shapley values for Graph Neural Networks models. It takes into account graph structure and node features in its explanation. It remedies the intractability of possible graph combinations by using a smart mask generator that reduces the number of nodes and features considered, informed by the architecture of the GNN models it's trying to explain.

There are also several methods that are adapted from Neural Networks, where they are widely used with text and image data, to Graph Neural Networks and that belong to the gradient-based category. A reference among these is GradCam [43], which is a gradient based method that computes a heat-map for each layer independently. This approach rely on computing the change in the gradients with respect to infinitesimal changes in the input features. As mentioned in Section 2.3 this gradient-based approach is an instance of the larger family of attribution methods and closely related to perturbation methods. GradCam is focused on node features explanations.

Decomposition-based methods decompose the prediction score in a series of terms associated to input features. In this manner they also allow for understanding the relations between input features and the output of the model. GNN-LRP [58] is an example of this approach.

So far, the highlighted works are adaptations for graph models of explainability methods that were previously used to explain predictive models for tabular, image and text data. The set of methods that follows were created from scratch for graph models and show increased consideration for the structural particularities of different graph models and data.

GNNExplainer [80] was the first explainer developed specifically for graphs, it combines node features and edge explanations. Explanations are collections of edges, but as far as it is understood, it works only for some types graphs and graph representations, not including for example relational graph variants.

SubgraphX [83] is another solution, it allows for true subgraph explanations, not just collections of edges, and it accomplishes this using the Monte Carlo Tree Seach (MCTS) method and reinforcement learning. As a downside, it has been reported to be generally less scalable than other approaches [82]. In

this case, the reduction in search space happens simultaneously with the search for the best explanation. From the approaches found in explainability for GNNs, SubgraphX belongs to a group that seems the most adaptable for KGs because of its main focus in edge explanations, and edges are more closely related to facts in KGs.

The previous two works are instances of perturbation-based approaches.

In another class of explanation methods it is relevant to mention a hybrid approach, where the authors were able to design a GNN in such a way that its results can be decoded into interpretable rules [11]. Despite its promising approach, this method is not general purpose since the GNN and the decoder are tailored only to each other.

3.3 Explainable AI for Self-Supervised Representation Learning

The methods mentioned so far are all feature-based, have explanations that rely on the understanding of an output in relation to some input feature. But if the input features are not interpretable, as is the case for the embeddings obtained from Knowledge Graphs and used in downstream Machine Learning tasks, the subsequent explanations will not be intelligible. In reality, in those cases it is possible to have one or two black-boxes since the embedding algorithm is opaque due to the resulting embeddings not being interpretable by humans and the downstream Machine Learning algorithm can also be black-box.

Several works have explored the possibility of having interpretable vectors by design. In EVE [47] the vector dimensions are defined beforehand as features based on Wikipedia articles and concepts. In INK [64] the learned vectors are binary feature-based representations of KG nodes where each feature can be for example a boolean value for a relation e.g. *is_capital_of*–False, or a pair relation–literal e.g. *founded_in*–1143. This work is related to the unsupervised generation of data mining features from Knowledge Graphs [41]. Still related to this, other more specific approaches consider interpretable features based on the field of application, like for example in the biomedical field [62].

The remaining option is to approach the explanation by perturbing the original input features, meaning the Knowledge Graph. In this case, some embedding methods are inductive, meaning that they adapt to new inputs but others are transductive (closed-form). In the latter case, an unseen instance of data, such as a perturbation of the original KG, does not have a corresponding embedding and it requires retraining the model. Retraining the model before new predictions is not in the scope of the methods discussed so far. All of them make intensive use of the model predict step, taking advantage of the easily accessible predictions to draw an explanation where they rely on repeatedly generating outcomes drawn from perturbations in the inputs. Taking into consideration the issue of perturbing closed-form embeddings and the heavy reliance of perturbation-based explanation models on efficient prediction steps, the time complexity of explanation methods for these cases is an additional and clear challenge.

A work that had previously tackled the issue of explaining Link Prediction is ExplaiNE [30]. In this work the issue of retraining for new embeddings is raised in relation to computational demands. Also, the problem of the existence of many local-minima that even for a single link removal may change the optimal embedding entirely is also addressed. In fact, what the solution does is to investigate the effect of infinitesimal changes to the weighted links between nodes, thus avoiding the previously mentioned problems. This work is focused on counterfactual reasoning, with two different modes of explanation,

the removal or addition of links in the graph, and correlating the importance of each removal or addition with the probability of the predicted link. However this work assumes that the graph is undirected and that can be represented by a weighted adjacency matrix.

The most promising solution in this case may be to consider alternatives to repeatedly retraining the model from scratch. In the topic of transductive embeddings for dynamic or evolving graphs, an efficient re-learning method is proposed for random walks based embeddings by EvoNRL [23] which relies on obtaining new network embeddings only in some situations that are deemed relevant and not everytime a unit change is made to the network. To accomplish that, this solution applies changes in the network not directly in the KG but in the set of random walks that were originally derived from the KG, this gives stability to the network representation while also being more efficient in dealing with changes in the network. It allows adding or deleting of edges or nodes. The authors of EvoNRL also argue that without this approach the results of subsequent training rounds would not be comparable. So, when comparing to a hypothetical retraining of the whole graph this approach avoids the instability characteristic of most embedding methods, especially random walk based [23]. This instability is studied for example by Borah *et al.* [4] mentioning especially Word2Vec in which RDF2Vec is based. Another approach to efficient re-learning could be achieved by known embedding algorithms that provide us with fast implementations as is the case with RDF2Vec Light [45].

In the area of explainability, Kelpie [55] also tackles the retraining of the embeddings as an essential part of the process in order to tie the explanations to the original model data in Knowledge Graph form. The major challenge that this work tries to solve is the time complexity of retraining for each perturbation, and explanation methods require many perturbations. They propose an alternative to retraining the whole graph called post-training, which consists in adding the new entity to an already trained model, while freezing other embeddings and parameters. In Kelpie, the original entity is unchanged, instead a new entity that is a copy of the original entity is added to the graph. If the entity is an exact copy it is called an homologous mimic, if the entity is a perturbed copy it is called a non-homologous mimic. This way, the attribution is extracted between the two mimics, resulting, according to the authors, in smaller fluctuations introduced by the post-training which ultimately lead to better results [55]. However, this work is limited to Link Prediction, and it does not assume downstream ML tasks, it just explains direct Link Prediction based on embeddings similarity.

In the context of Link Prediction, Kelpie [55] also reduces the computational complexity of finding a solution by accomplishing a reduction of the initial search space with heuristics to find the most promising explanations. They do this by leveraging graph topology, stating that to explain some prediction $< h, r, t >$, taking some potentially explaining fact $< h, r, q >$, the closer q is to the predicted entity t the most promising is the fact $< h, r, q >$. They limit the search space to some top-k facts found this way and sorted by a promisingness value. However, this solution does not translate to Node Classification because in principle, the relations that lead to a prediction in Node Classification can be farther away in the graph, and most importantly, the predicted class is not part of the graph in most applications.

In a setting similar to link prediction, SEEK [63] is concerned with a relation prediction task between two entities where the entities are encoded using a KGE model and the relation is predicted using a second model within a Supervised Learning setting. Due to the biological setting, the goal of this work is to explain the relation prediction by finding the most important biological semantic aspects that are

shared by the two entities in the KG. Instead of learning a direct representation of an entity, this work represents an entity as an aggregation of its semantic aspects. Similar to the previous approaches, the explanation approach is again perturbation-based. In this case the perturbations are applied for example by removing from the original representation the embeddings for some shared semantic aspect of the two entities whose relation is being explained.

According to the conducted literature review, only one work developed an approach that tries to explain the results of a common graph embedding method, namely node2vec [18]. Its unique feature is that it does not explain predictions, but it explains the embeddings themselves. The main principle is to create a perturbed graph where the perturbation is materialized in the weakening of edges. Then the authors leverage EvoNRL [23] to perform an efficient re-learning were only the renewed walks are used to fine-tune the neural weights and thus the embeddings. This explainer by Kang and Park [31] is based on SubgraphX [83] and it struggles even more with scalability. Moreover, the work is limited to node2vec homogeneous graph embeddings, so it is not applicable for Node Classification in a KG, and it has some shortcomings in regards to the established baseline and the validation method used. Since no code is available, the replication of this work is much hindered and the reliability of the results cannot be easily confirmed.

None of the aforementioned works have dealt with Node Classification using KGE, in this step, it is critical to adapt the existing component to integrate the downstream ML task actually responsible for the Node Classification, thus enabling a solution with the ability to explain end-to-end Node Classification tasks involving KGE as is the proposed goal for this work.

3.4 Other Topics in Explainability

3.4.1 Counterfactual Explanations

The types and form of explanations are another major concern in the area and may contribute or hinder the human understanding of the provided explanations.

There are considerable studies in psychology and cognitive science that show that humans have a natural predisposition to reason in counterfactuals [7]. According to Guidotti [19], counterfactuals are one of the most valuable approaches to generate human understandable explanations in the context of XAI and there has been an explosion of works that apply counterfactual explanations.

According to Yeh *et al.* [79] most post-hoc explanation can be identified under one or more of the following categories: feature-based explanations in which the model output is related to the input features, example-base explanation methods in which the model output is related to the training samples, and counterfactual explanations. Counterfactuals answer the question of "what would have to change in the input in order to cause a different result", or put in a different way, "what could have happened differently if the input was different in this or that regard".

Several authors divide counterfactual explanations in two modes, necessary and sufficient [38, 55, 76], but in practice it seems there are other terminology closely related and that in practice seems to be accomplishing the same thing, for example, for some authors, a necessary explanation is a contrastive explanation with pertinent negatives, and a sufficient explanation is a contrastive explanation with pertinent positives [19]. For simplicity, considering the definition from propositional logic, x is a necessary

condition for y iff $y \longrightarrow x$, and x is a sufficient condition for y iff $x \longrightarrow y$. This approach seems to provide a more comprehensive explanation than most methods since it usually employs in a single solution two different ways of extracting knowledge from the explanation.

3.4.2 Evaluation

Especially when considering methods applied to KGs, the lack of ground-truth datasets is one important issue. Concerning ground-truth datasets, for the Link Prediction task only in the work of Halliwell *et al.* [20] it is found a proposal of a dataset that includes ground truth explanations. Even if such work existed, it would not be enough to validate the results against a single dataset. According to the conducted literature review, no ground-truth is available for Node Classification on Knowledge Graphs which is why none is considered in this work.

To circumvent this limitation the authors of ExplaiNE [30] derive ground-truth explanations for their datasets using metadata. It consists in formalizing some intuitions about the data and the predictions being made, validating those assumptions with the help of statistical tests, and consider the validated assumptions as ground-truth explanations. These groud-truth explanations were obtained for Link Prediction tasks only, and in simpler homogeneous graphs, and are not reusable for Node Classification using KGs.

Kelpie [55] uses as evaluation metric an effectiveness measure which consists in quantifying the difference in some Link Prediction metric, e.g. $H@1$, between the results of the original prediction model and the results of the model when the explaining facts were removed from the dataset. SEEK [63] uses a similar approach. The concept of comparing the results of an original and modified graph had also been used previously in EvoNRL [23].

Chapter 4

Methodology

This chapter is mainly concerned with presenting the methods and solutions that constitute the two proposed explainability solutions, LoFI and C-KEE. Section 4.1 presents the description of the explainability approach that is common to both solutions. Sections 4.2 and 4.3 address the specific methods and solutions used in LoFI and C-KEE, respectively. Section 4.4 gives details about relevant explainability measures. Additionally, the procedure used to evaluate the solutions is addressed in Section 4.5 and the pipeline that describes its development and application is described in Section 4.6.

4.1 Explanation Approach for Node Classification with KGE

The methods employed focus on two main issues, one is to generate explanations in a timely manner so that the solution is fast enough for practical applications and this is closely related to the retraining issue intrinsic to the transductive embedding methods being explained, other is to produce relevant explanations with a thought about human thinking and cognition.

Perturbation Methods on Knowledge Graph Embeddings

The major approach behind LoFI and C-KEE is to apply an attribution-based method by implementing a perturbation-based approach. This follows one of the most successful trends in Explainable AI with popular solutions in both general purpose frameworks as well as more focused explanation methods in all kinds of problems.

The developed explanation methods provide example-based explanations by selecting particular instances present in the dataset (particular facts or entities), to explain some prediction. Intuitively, this makes much sense in the context of Knowledge Graphs as sets of real-world instances. This example-based approach and the pool of instances that are effectively considered for a potential explanation enables tractable complexity to find influential instances.

Counterfactual Explanations with Necessary and Sufficient Conditions

Regarding the type of the explanation, the proposed explanation methods are counterfactual-based methods that employ contrastive information to answer what-if questions.

For LoFI, an explanation is defined as the set of the most relevant facts identified as necessary or sufficient to achieve a specific outcome. For C-KEE the explanation consists in sets of the most relevant

neighbours. These explanations are considered counterfactual in nature. Necessary explanations support answering the question *"what would happen to the entity classification if it was not defined by these facts (or neighbours)?"*, whereas sufficient explanations aim to answer *"what would happen to the entity classification if it was only defined by these facts (or neighbours)?"* These explanations are more formally defined in Sections 4.2.2 and 4.3.1, for LoFI and CKEE, respectively.

To determine a successful explanation, an explanation condition must be met. The explanation methods were designed with two conditions in mind: the *class probability condition* and the *class change condition*. The *class probability condition* is based on the prediction probability of the originally predicted class dropping (or not) below a certain threshold for a successful necessary (sufficient) explanation. The more strict *class change condition* requires the predicted class to change for a successful necessary explanation or remain the same for a sufficient explanation.

4.2 LoFI Explanation Solution

This section presents the core components of the implemented explainability solution, designated Local Facts for Interpretability in Node Classification with Knowledge Graph Embeddings, or LoFI for short.

4.2.1 Perturbations in LoFI

The goal is to provide explanations for the coupled Knowledge Graph Embeddings and Node Classification (KGE+NC) model, and because of this the perturbations are thus applied to the input of the KGE model, meaning the Knowledge Graph, and their impact is measured in relation to the output of the Node Classification model which is a Supervised Learning model. Since the Knowledge Graph is a collection of facts, the solution is in fact applying an example-based approach in which a unit perturbation consists in the removal or addition of a single fact from the Knowledge Graph. The principle is simple, remove or add a fact and measure the impact that such modification has on the performance of the model. This is the most straightforward approach for perturbations in KGs because contrary to weighted graphs it's not possible to apply a small change to the weight of the edge.

Because of the transductive nature of the KGE models of interest for this work, any perturbation of the original KG implies that the KGE should be retrained from scratch. So, to generate a true perturbation, the models would have to be retrained again which in fact means that they would obviously cease to be the same model that was to be explained in the first place. So, what LoFI accomplishes is a solution that avoids the need for a full retraining by employing an update method that approximates the effects of the perturbation, thus generating explanations much faster and without relying on a completely new model.

4.2.2 Counterfactuals in LoFI

The necessary and sufficient explanations in LoFI are defined as:

Definition 3 - Counterfactual Necessary and Sufficient Explanations.

- A necessary explanation for a given entity e is the smallest set of facts $< e, r, t >$ that, when removed from the KG, satisfy a condition C_{nec} for necessary explainability; and
- A sufficient explanation for a given entity e is the smallest set of facts $< e, r, t >$ that, when kept in the KG, satisfy a condition C_{suf} for sufficient explainability.

For the necessary explanations, the initial representation of the entity has all the original facts and the algorithm implements the process of removing a fact. For the sufficient explanations, the initial representation of the entity is empty and the algorithm implements the process of adding a fact.

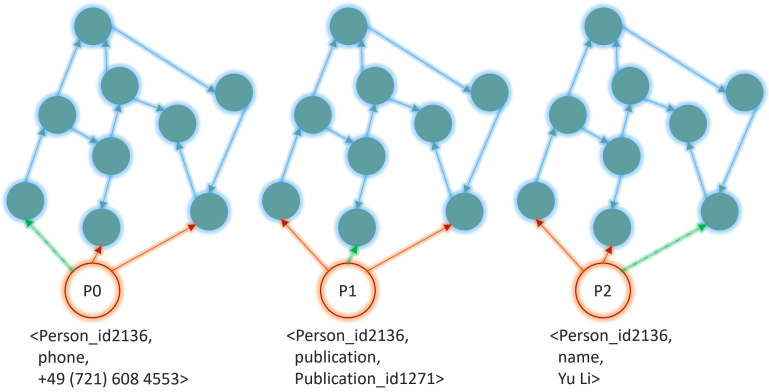

Figure 4.1: Diagram illustration of the possible single perturbations to an entity P on the KG, where for a necessary perturbation only the light green edge is removed and for a sufficient perturbation only the light green edge is kept.

Figure 4.1 shows an example for an entity to explain *Person_id2136* where for that entity each light green arrow represents an edge or fact that is also labelled below each graph. For necessary explanations each graph is a different perturbation where only the light green edge is removed. For sufficient explanations each graph is a different perturbation where only the light green edge is kept.

4.2.3 Explanation Methodology

The intuition behind LoFI consists in the idea of what would happen to the predictive model if some fact or facts, linked to an entity that is to be explained, were removed from the KG. To accomplish that purpose LoFI uses online learning methods to update the embeddings "as if" the original KG had been modified by the removal of that fact or facts (for necessary explanations). The implementation of the method is concerned with optimizing those perturbations for embedding methods based on random-walks, but with some modifications it could be argued that the approach also works for other kinds of embedding methods.

A summary of the LoFI explanation method for explaining a single prediction instance is summarized in the following paragraphs with a step-by-step description, using Figure 4.2 for support with the steps marked and labeled.

The preliminary condition (*step 0*) is to take the Knowledge Graph and labeled entities and train the KGE+NC prediction models as usual in any Node Classification application. The trained KGE model

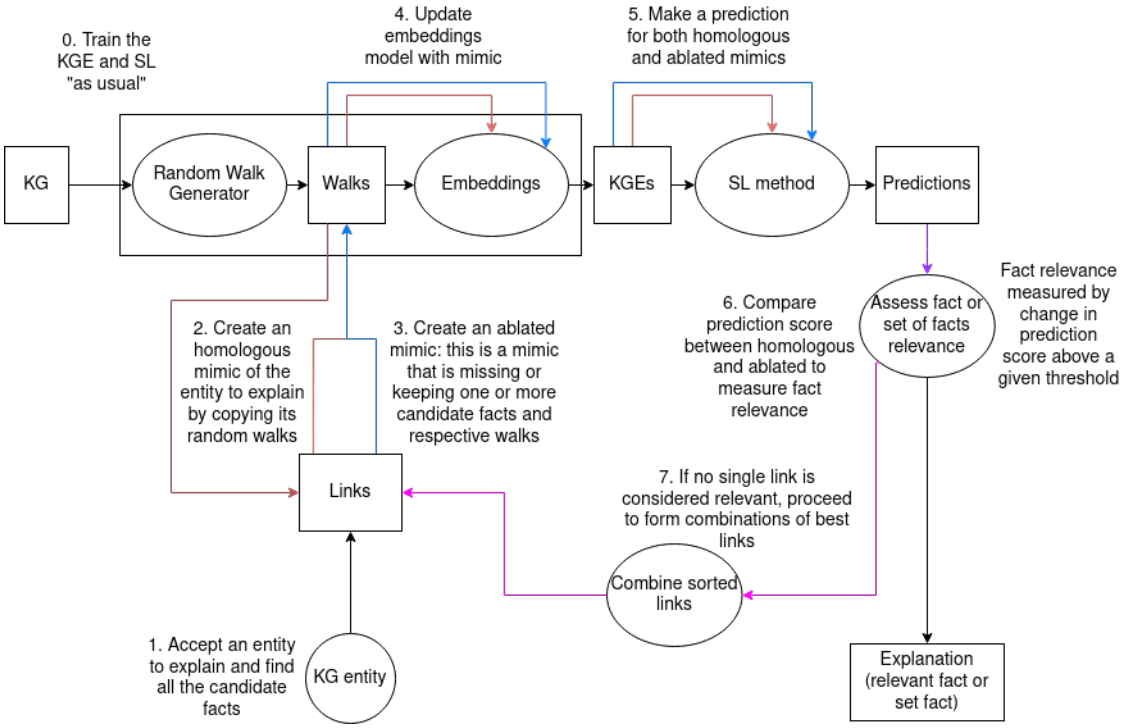

Figure 4.2: Overview of the LoFI explanation method. This diagram exemplifies the application of LoFI to explain the classification assigned to some entity.

and the trained NC model are given as inputs to the explanation framework, and in the KGE model there is direct access to the extracted walks. The steps that constitute the explanation pipeline are described as follows:

1. The method accepts an entity to explain and finds the candidate facts for that entity (1-hop facts);
2. Using the walks extracted from the KGE model, find the walks where the entity to explain participates and create one homologous mimic by making an exact copy of those walks;
3. Using the same walks from *step 2* create n ablated mimics, one for each candidate fact by removing or keeping the candidate fact according to the type of explanation, if it's necessary or sufficient, respectively;
4. Independently update the embeddings model with each mimic and respective walks to extract the mimic embeddings;
5. Using the predictive step of the SL model, output the predicted class probabilities and predicted class for the homologous mimic and ablated mimics. This requires a model able to output probability scores;
6. Using the class probability condition: compare the scores between the homologous mimic and each ablated mimic and take their difference as a measure of relevance. If the relevance score is higher/lower than the predefined threshold then the candidate fact used to generate that the ablated mimic is considered a relevant necessary/sufficient explanation, respectively, for the entity to explain.

7. If no ablated mimics were considered relevant in the previous step then the method proceeds to use combinations of candidate facts to generate new mimics and the process is repeated.

The type of explanation is a parameter for the explanation, either the pipeline runs for necessary or for sufficient explanations. To explain any number of explanations the explanation pipeline is used that many number of times.

4.2.4 Facts in 1-hop Neighbourhood

Considering an entity as an origin point in the graph, a k-hop fact is a fact that connects the origin entity to an entity k hops away using the shortest path between the origin and the k-hop entity. A 1-hop fact is a fact that connects the origin entity to an entity 1-hop away with the origin. A 1-hop fact has the origin entity in its constitution.

From an interpretability viewpoint, LoFI considers that an entity is primarily understood by the set of facts that mention said entity, or in other words, the facts that define the understanding of an entity are the facts that have that entity as one of the constituents of the triplet, either as head or tail. An implicit assumption is that the entity is semantically defined by these facts which seems a very reasonable assumption since these facts are truly what enables the connection of the entity to the rest of the graph. So, the facts that promote the understanding of the entity are the 1-hop facts for that entity. For example, in the aforementioned Figure 4.1 the entity *Person_id2136* is defined by three 1-hop facts.

Another consequence of considering only 1-hop facts is that the search space for the solution is limited to a tractable problem when compared to evaluating all the facts in the k-hop neighbourhood, where the choice of k could result in a combinatorial explosion of facts to evaluate.

The candidate facts for explaining some entity is the set of facts that are evaluated as explanations for the prediction associated to that entity. In the explanation pipeline, all the 1-hop facts associated with the entity to explain are candidate facts.

4.2.5 Substitute Method for Perturbations in the Knowledge Graph

One of the main challenges in applying the most popular explanation methods to KGE is the static or transductive nature of such embeddings, in which all the entities must be present at training time. It is then a closed representation of a Knowledge Graph admitting no changes. For the purpose of perturbation based methods this is very limiting because technically it requires the retraining of the whole graph for each desired perturbation, which is unfeasible. To overcome this issue is one of the core challenges of this work.

Translating Perturbations into Walk-based Embeddings

LoFI was specifically developed for explanations where the KGE models are walk-based. One of the reasons was to prioritize these popular solutions because of their successful results in Node Classification [44]. Also, the ability to directly modify the walks provides an efficient solution to implement the perturbations for the update method and enables some added stability in the representation since it reuses the walks of the original model. The use case for this work are random-walk based embeddings obtained with RDF2Vec.

Walk-based solutions for KGE are usually composed of two separate steps. The first step is to take the KG as input to generate a set of walks. These extracted walks are taken to be themselves actual representations of the Knowledge Graph. The second step is to take the set of walks to generate the embeddings.

With a design like this, the procedure of removing or adding a fact to the KG could be accomplished in more than one way. The first solution is one where the removal or addition could be done directly in the KG thus obtaining a new and perturbed KG. The second solution, and the one used in this work, is to implement the removal or addition of facts directly in the walks themselves. Because everything that the embedding step uses is the set of walks, instead of defining the 1-hop neighbourhood using the Knowledge Graph, it can actually be defined in practice by extracting the 1-hop facts that are actually present in the walks themselves. If, for example, some fact was present in the KG but is not present in the walks, then such fact is actually not part of the model so it can be ignored in any case. Such fact has no effect on the model and would not contribute to any meaningful explanation.

By using the already available walks, several benefits were achieved. Some of the benefits of this approach are that it is more straightforward and faster to modify directly the set of walks as opposed to modify the Knowledge Graph. Also, it is faster to obtain the perturbed walks directly from the original walks as opposed to having to extract a new set of walks based on a new KG were the fact was removed or added. Also due to the intrinsic randomness of the walks extraction process, a new set of walks might end up representing the same entity in a different way so that the two would not even be comparable, so by directly modifying the walks it is possible to obtain a representation that is closer to the original entity. For the purpose of explainability, and when comparing to a hypothetical retraining of the whole graph this avoids the instability characteristic of most embedding methods, especially random walk-based, to which RDF2Vec belongs.

At this point, it's also important to recall that, in the most common approaches, each walk in the set of walks that characterizes an entity always starts at that entity. Because, as defined before, the explaining facts for some entity are to be retrieved from the pool of 1-hop facts of that entity (the candidate facts), then each walk is guaranteed to have at least (and usually at most) one of those 1-hop facts.

Also, the candidate facts for some entity can show up at different levels in some specific walk and it can show up in the walks of other entities. To implement the perturbation:

- For necessary explanations it implies that every walk, where the candidate fact is present, is truncated prior to that fact; and

- For sufficient explanations it implies that for every walk that belongs to the explained entity, where the candidate fact is not present at the beginning of the walk, is removed.

The above removal process is applied at all the levels where a candidate fact can show up in the set of walks as described before.

This solution should work for any walk-based KGE solutions, although in practice it was implemented for the case of KGE extracted using the RF2Vec solution.

Update Methods

Incremental learning procedures are algorithms very common in ML solutions. They arise from the need to evolve a model that was learned before to enrich it with new information and patterns without forgetting its existing knowledge. Despite the fact that KGE are transductive in nature, there have been attempts to circumvent that limitation and make such algorithms allow a more dynamic behaviour. RDF2Vec [53] uses Word2vec [36] to train the embeddings for which a solution is readily available in the form of an Update method implemented in the most popular open-source Word2vec package[1].

The update method is essential to LoFI because it uses incremental learning procedures to circumvent the need for a complete retraining from scratch each time a perturbation is applied. Update methods may have negative impact on the already learned model if not used appropriately and this is especially true for transductive learning. For this reason, the current work aims to minimize any harmful effects by keeping changes to a minimum. In relation to the original model there is only one single event of change from t_0 (the original model) to t_1 (the perturbed model) and one single entity being added (the perturbed/ablated entity) and the original embeddings do not suffer any change at all. At this point the perturbed/entity is more correctly designated the perturbed/ablated mimic because it's now clear that it's not the original entity that is being perturbed but only a copy (or mimic) of that entity.

For an update method to work with LoFI there is need for it to accept as input a:

- New entity to train;
- New set of walks that concern the new entity to train;
- Parameter: option for new training parameters (epochs, start learning rate, final learning rate, and others depending on the model) - these parameters may be tweaked to enable an homologous mimic with higher similarity to the original entity;
- Parameter: option to freeze all the other embeddings during training, designated for future reference *freeze_embeddings* - to avoid any changes to the original embeddings during the update procedure; and
- Parameter: option to initialize the new entity to train with the initialization used in one of the original entities, designated for future reference *init_mimic* - to reduce the variability caused by a random initialization.

In the update method of Word2vec there was no option to initialize a new entity according to the needs of this work so the source code of Word2vec was modified to accomplish this requirement.

The perturbation and update step is illustrated inside the dashed line in Figure 4.3. Conceptually, and for simplicity, the perturbations are considered to happen in the KG, although in practice they are applied using the set of walks that step is omitted here. On each graph the original entity O is displayed along with a perturbed/ablated version M_i. For each perturbation, the update method consists simply in taking the original KGE model and the original entity, adding the ablated entity to the KGE model and perform the incremental learning.

[1] `https://github.com/piskvorky/gensim`

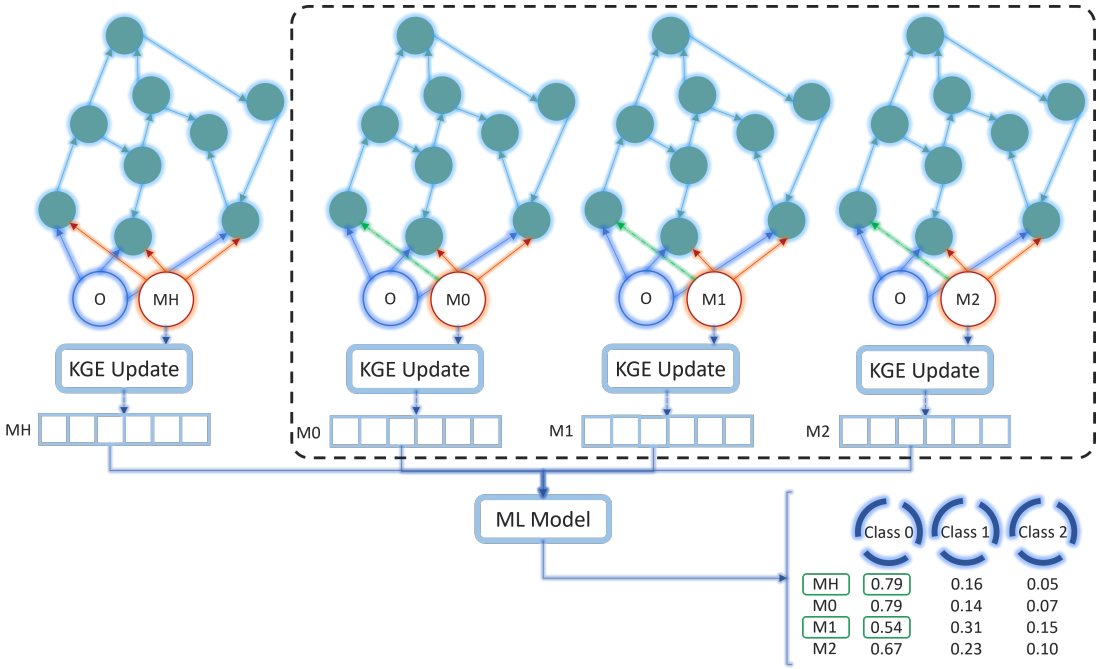

Figure 4.3: Overview of the mimic entities and update method used in LoFI. The diagram illustrates how the homologous mimic and ablated mimics are generated and used to obtain their associated predictions.

To approximate the behaviour of the mimics to the original entities, an optimization of the aforementioned update method parameters may be used, consisting on a grid search for the parameters that result in a better approximation of the homologous mimic to the original entity. The approximation is measured using the cosine distance measure of similarity and the agreement between predicted classes.

Homologous Mimic and Ablated Mimics in Update Methods

Regarding the concern that training an entity with incremental learning is not the same as training it during the original training procedure, and that such behaviour would impoverish the results of comparing an original entity trained in the original model with the ablated mimic trained with the update method, the concept of homologous mimic was used to improve the comparison logic.

An homologous mimic is simply an exact copy of the original entity that is added in the incremental learning. As illustrated in Figure 4.3 on the graph on the left, alongside all the ablated mimics, an additional entity is considered, a mimic that is an exact copy of the original entity but that goes through the update procedure. The comparison logic is therefore to compare perturbed entities with the homologous mimic because the conditions for their training are much more similar.

4.2.6 Generating Compound Explanations

In the case where the acceptance criteria for a good explanation is not fulfilled by any of the candidate facts there are two fallback solutions, one is to take the fact with the highest relevance score as the best possible explanation, the other is to allow for multi-fact explanations with the combination of those same facts in sets of increasing size. Here, the maximum length of explanation is defined as the maximum amount of facts that should be used in the explanation. The attribution of each fact is then computed and evaluated, incrementally starting at explanations of size 2, up to the maximum length or until the

acceptance criteria is reached. With this the aim is to obtain true explanations that take into account the inner workings of the models to explain. Considering that synergistic effects can exist between combined facts, it is not enough to simply combine the obtained relevance of the two or three, etc., best facts to output an explanation. The proposed approach can be considered a true subgraph explanation.

This also means that the combinatorial explosion of testing every fact with every other fact is not practical when considering tens or hundreds of candidate facts. In the case where multi-fact explanations are applied, a greedy method is also applied where a sorted list of the previously found best facts is obtained and then the top k facts are combined between themselves to generate new test cases of aggregated facts.

4.3 C-KEE Explanation Solution

This section presents the core components of the implemented explainability solution, designated Classification with Knowledge Graph Embeddings Explained (C-KEE).

4.3.1 Counterfactuals in C-KEE

The necessary and sufficient explanations in C-KEE are defined as:

Definition 4 - Counterfactual Necessary and Sufficient Explanations.

- A necessary explanation for a given entity e is the smallest set of neighbours N that, when having their embeddings removed from e's representation, satisfy a condition C_{nec} for necessary explainability; and
- A sufficient explanation for a given entity e is the smallest set of neighbours N that when having their embeddings used to represent e, satisfy a condition C_{suf} for sufficient explainability.

For the necessary explanations, the initial representation of the entity starts with all the neighbours. For the sufficient explanations, the initial representation of the entity is empty.

4.3.2 Explanation Methodology

Figure 4.4 presents an overview of the C-KEE approach. In the first step, embedding representations for each KG entity are learned using a KGE method. C-KEE is agnostic to the specific KGE method employed. In the second step, *Global predictions*, representations for each KG entity are generated by aggregating the individual embeddings for each of its direct neighbours (i.e., the tail nodes of the triples it participates in). These global representations are then used to train a supervised learning model for node classification. In the third step, *Explanation generation*, two types of explanations, *sufficient* and *necessary*, are created by identifying for each classified node the sets of neighbours that affect the global supervised model prediction. The necessary explanations provide insights into which neighbours are necessary for a particular classification to be made, while the sufficient explanations reveal the ones that are sufficient to support a particular class assignment. Each explanation identified by C-KEE is assigned an explanatory power score that is based on the impact observed in classification.

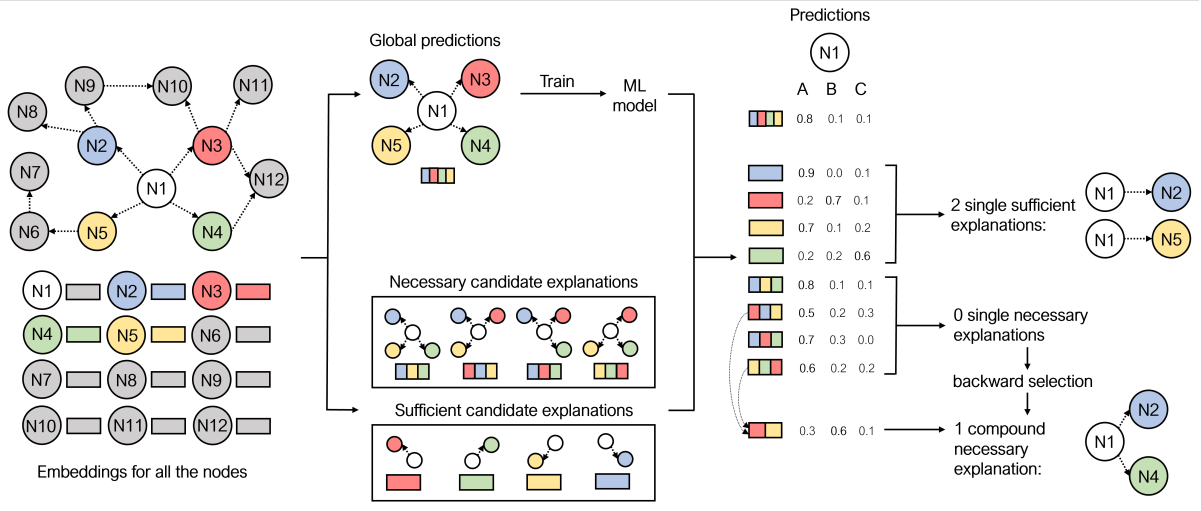

Figure 4.4: Overview of C-KEE. This diagram exemplifies the application of C-KEE to explain the classification assigned to entity *N1*, using both sufficient and necessary explanations.

4.3.3 Entities in 1-hop Neighbourhood

From an explainability viewpoint, C-KEE considers that an entity is primarily understood and semantically defined by the set of facts that mention it. By extension, this means that an entity is defined by its neighbours.

Contrary to most common applications of KGEs in node classification, where the entity is represented by the embedding vector of the entity itself, in C-KEE, the entity is represented by the embeddings of its neighbours. This aggregated representation is a key aspect that allows it to circumvent the need for retraining or updating the KGE model when triples are added or removed from an entity's representation. Any aggregation operation that preserves embedding size can potentially be used, and these experiments employed the mean. When an entity is represented by all the embeddings of its neighbours, it is considered a global representation. The global representations are used as input features to train the ML model for node classification.

4.3.4 Explanation Generation

C-KEE generates two types of explanations: necessary and sufficient. Both algorithms require the definition of a maximum explanation length (i.e., the maximum number of neighbouring triples accepted in an explanation) and of the explanation condition (i.e., class change or probability change with associated threshold).

In the following, a necessary explanation generation is used as an example. To explain a classification outcome for a given entity e, the first step is to take the original prediction made on its global representation. Then, for each neighbour of e, remove its embedding from e's representation to generate a candidate explanation and apply the classification model using the ablated representation. If the explanation condition is met, the corresponding candidate explanation is considered valid. When all neighbours have been evaluated if at least one valid explanation has been found, the procedure finishes and outputs all valid explanations found. If not, then a backwards greedy selection is employed where the neighbour with the highest relevance (i.e., that in which removal resulted in the largest difference between the original class

Algorithm 1 Generate necessary explanations.

Input: entity e;
KGE model K;
node classification model M;
explanation maximum length l;
Output: the set X of explanations

$X \leftarrow \emptyset$
$relevance_{best} \leftarrow 0$
$N \leftarrow$ ALL NEIGHBOURS(e)
$embs \leftarrow$ GET NEIGHBOUR EMBEDDINGS(K, N)
$repr \leftarrow$ AGGREGATE REPRESENTATION$(embs)$
$class_o, class\_prob_o \leftarrow$ PREDICT$(M, repr)$
$curr\_len \leftarrow 0$
$explanation \leftarrow []$
$best\_explanation \leftarrow []$
while $X == \emptyset$ and $curr\_len <= |N|$ and $curr\_len <= l$ **do**
  for $neighb \in N \backslash best\_explanation$ **do**
    $embs' \leftarrow embs.remove(best\_explanation \cup neighb)$
    $repr' \leftarrow$ AGGREGATE REPRESENTATION$(embs')$
    $class_x, class\_prob_x \leftarrow$ PREDICT$(M, repr')$
    $relevance \leftarrow$ COMPARE$(class_prob_o, class_prob_x)$
    $explanation[curr\_len] = neighb$
    if $class_o \neq class_x$ **then**
      $X.append(explanation)$
    else if $relevance >= relevance_{best}$ **then**
      $past\_explanation = explanation$
      $relevance_{best} = relevance$
  $best\_explanation = past\_explanation$
  $curr\_len += 1$
if $X = \emptyset$ **then return** $best\_explanation$
else return X

probability and the novel one) is combined with all other neighbours to produce ablated representations missing two neighbours. This iterative process continues until an explanation is found, all neighbours of e have been evaluated or the maximum explanation length is reached. If the end of the procedure is reached without an accepted explanation, then the explanation with the highest relevance is taken as the best possible explanation. This algorithm ensures that explanations are minimal in length, but it does allow for multiple explanations of equal size that meet the condition criteria to be found. Algorithm 1 presents this approach using the class change condition, which can be trivially adapted to use the class probability condition.

The process is analogous for sufficient explanations (see Algorithm 2 in Appendix C), but it uses forward selection, whereby single neighbours are first evaluated as representations and then combined iteratively until an explanation is found, all neighbours have been considered, or the maximum length is

reached.

The time complexity of the explanation search is $\mathcal{O}(kn) \sim \mathcal{O}(n)$ where k is the length of explanation and n is the number of neighbours, since $k \leq 5 \ll n$ in our work.

4.4 Explainability Conditions and Measures

Relevance Measure

The relevance measure quantifies the importance that a neighbour or set of neighbours in C-KEE (fact or set of facts in LoFI) have for explaining some prediction, it is the core measure used to compare candidate explanations between themselves and also relative to a predefined threshold of what is a good explanation. To be able to compute the relevance measure, the Node Classification model has to be able to output the prediction probabilities for each class. In this manner an attribution is created between each relevance measure and the neighbour or set of neighbours in C-KEE (fact or set of facts in LoFI) that were removed from the candidates of the entity to explain.

The relevance measure is, taking the output of the SL model, the difference between the prediction probability for the predicted class of the global representation in C-KEE (homologous mimic in LoFI) and the prediction probability for the same class of the ablated representation in C-KEE (ablated mimic in LoFI). For the purposes of this work, this means that:

- For necessary explanations, the larger the relevance value the better, because it means that the removed neighbour in C-KEE (fact in LoFI) caused the prediction probability to decrease the most possible.
- For sufficient explanations, the smaller the relevance value the better, because it means that the kept neighbour in C-KEE (fact in LoFI) caused the prediction probability to decrease the least possible.

Explanation Threshold

The explanation threshold must be defined in order to implement the class probability condition. The threshold for a good explanation is defined based on the performance of the classification model. The threshold is defined based on the dispersion of the classification model, which is taken as the standard deviation observed in the model performance for the predicted classes. This requires that the ML model is implemented with a method returning predicted class probabilities.

Explanatory Power

Each explanation generated is assigned an *explanatory power*, which aims to be an intuitive measure of how well the explanation captures the node classification. The explanatory power is calculated based on the class probabilities assigned by the ML models used for node classification.

The intuition behind the explanatory power of a necessary explanation is that the lower the probability assigned to the class is, the highest the explanatory power is, since removing the explanation neighbours from the entity representation in C-KEE (removing the explanation facts from the KG in

LoFI) implies a more pronounced effect in the classification. Conversely, the explanatory power of a sufficient explanation reflects that the smaller the loss in classification probability observed when using the explanation neighbours in C-KEE (explanation facts in LoFI), the better they are at explaining the prediction. The following equations define these measures:

$$xpower_{nec} = (class\_prob_o - class\_prob_x)/class\_prob_o. \tag{4.1}$$

$$xpower_{suf} = 1 - (class\_prob_o - class\_prob_x)/class\_prob_o. \tag{4.2}$$

where $class\_prob_o$ is the predicted class probability of the original entity, and $class\_prob_x$ is the predicted class probability of the entity after removing (keeping only) the explaining neighbours in C-KEE (explaining facts in LoFI) for a necessary (sufficient) explanation.

4.5 Evaluation

4.5.1 Effectiveness Metrics

The evaluation of the performance of the explanation leverages the most common metrics for the performance of Node Classification models, accuracy and F1-score. The F1-score is weighted due to the imbalance of classes present in all the datasets considered. It balances the importance of majority and minority classes but without giving too much importance to the minority classes as it would happen with a Macro F1-score. The idea is to have a balanced metric that values relevant explanations across all classes.

In order to compute the evaluation metrics, the performance of the global representation (in C-KEE) or original model (in LoFI) is compared to the performance of the model when the explaining neighbours (in C-KEE) or facts (in LoFI) are removed or added. This is materialized in taking the difference between the metrics before and after a perturbation.

In C-KEE, for each entity to explain, the performance of the model with the chosen neighbours removed is already available during the explanation generation, as is the performance of the global representation. So, to compute the effectiveness metrics is just a matter of applying the effectiveness metrics to the available scores for all the explained entities. The metrics can be defined as:

Definition 5 - Effectiveness Metrics for C-KEE.

ΔAccuracy/ΔF1-Score is the difference between the Accuracy/F1-Score of the global representations for all the *entities to explain* and the Accuracy/F1-Score taken from the aggregated predictions of each individual *entity to explain*, where each *entity to explain* is the ablated representation using the selected explaining neighbours.

In LoFI, for each entity to explain, the chosen facts are removed from the original KG, the KGE+NC models are retrained from scratch using the same learning model, and the new prediction probabilities for that entity to explain are extracted. Then the prediction probabilities taken this way for all the entities to explain are aggregated and the performance metric is calculated. Finally, the difference between this metric and the original metric is calculated. Because all the models are trained from scratch for each entity, the evaluation stage is a computationally intensive step. The metrics can be defined as:

Definition 6 - Effectiveness Metrics for LoFI.

ΔAccuracy/ΔF1-Score is the difference between the Accuracy/F1-Score of the original KGE+NC model for all the *entities to explain* and the Accuracy/F1-Score taken from the aggregated predictions of each individual *entity to explain*, where for each *entity to explain* the KGE+NC model is retrained from scratch under the exact same conditions as the original KGE+NC model but where the original KG has been modified by removing or keeping the explanation fact or facts.

4.5.2 Comparison with Random Explanations

The goal is to take the explanations obtained with the proposed explainers and compare them with explanations randomly sampled from the pool of candidate neighbours (for C-KEE) or facts (for LoFI). Ideally, there will be a difference between the two explanations when evaluated, proving the usefulness of the proposed explainers.

To make the comparison as fair as possible, the random explanations are randomly sampled from the same pool of candidates as the explanations, except that the relevant neighbours (for C-KEE) or facts (for LoFI) are removed from the pool. The goal is to prove that the found explanations are effectively better then the other possible explanations, whatever those might be.

Because there is no way of determining how many neighbours/facts to sample for the random explanations, the same number of neighbours/facts that make up the explanation for some entity to explain is used for the number of samples of random neighbours/facts for that entity to explain.

The effectiveness metrics are applied to explanations as well as to random explanations which means that, for each entity to explain, in LoFI the models are again retrained from scratch. This further intensifies the time spend in the evaluation phase for LoFI.

4.5.3 Comparison with End-to-End Prediction and Explainability using GNNs

Given the originality of the proposed work, according to the conducted literature review there is no explainability method dedicated to explain Node Classification tasks based on KGE that could serve as a true baseline. Therefore, the results are also compared to the use case most close to this work in terms of application: selected popular GNN explainers are applied to GNN predictive models where the same Node Classification tasks are being solved.

Table 4.1: GNN explainer methods summary. E: edge-based, N: node-based, NF: node feature-based.

Method	Class	Type
Grad [60]	Gradients	E/N
GNNExplainer [80]	Perturbations	E/NF

The methods to be used were chosen taking into account popularity, performance and availability. Other requirement was to have representative methods of different explainability approaches. Recalling the taxonomy of explainer methods, Table 4.1 summarizes the selected methods regarding the taxonomy

class of origin and the form of the explanation obtained by the method. The implementations for these methods are taken from the BAGEL [50] benchmark available online[2].

Both these methods output a relevance score for each edge in the graph associated to some prediction. To enable a comparison to the explainers proposed in this work, the top scoring relevant edges were taken as necessary explanations and the same effectiveness metrics evaluation procedure was employed were the models were retrained from scratch but without the found relevant edges.

4.6 Development and Evaluation Pipeline

4.6.1 Code Base

The implementation of the LoFI explainer model was accomplished using, as a starting point, the open-source code base for the Kelpie framework [3] associated with the work from Rossi *et al.* [55] which was extensively modified for this work's purposes. The implementation of the C-KEE explainer model was accomplished using, as a starting point, the open-source code base for the SEEK method [4] associated with the work from Sousa *et al.* [63] which was extensively modified for this work's purposes.

4.6.2 Prediction Models Training and Evaluation

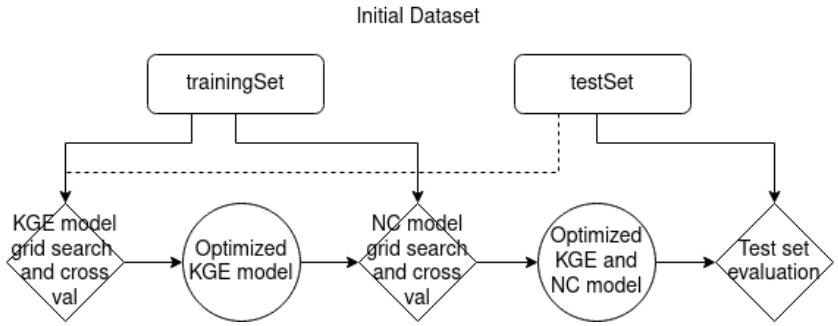

Figure 4.5: Training pipeline for the KGE+NC model.

To establish the required Node Classification models whose predictions are to be explained, a ML training pipeline was implemented. The pipeline is mainly focused in the training, optimization and validation of the KGE+NC model. An overview of the process is given in Figure 4.5 and is described as follows:

1. The KGE model is optimized using a grid search and validation approach with stratified 5-fold cross validation. This step allows the selection of the optimized parameters for the embeddings model and the selection of the base SL model (model with default parameters);

2. A KGE model trained with optimized parameters is obtained;

[2] https://github.com/Mandeep-Rathee/Bagel-benchmark
[3] https://github.com/AndRossi/Kelpie
[4] https://github.com/liseda-lab/seek

3. Using the embeddings from the KGE model, the NC model is optimized using a grid search and validation approach with stratified 5-fold cross validation. This step allows the selecion of the optimized parameters for the Node Classification model;

4. A KGE+NC model trained with optimized parameters is obtained; and

5. The final KGE+NC model is evaluated using the test/holdout set.

4.6.3 Development Pipeline

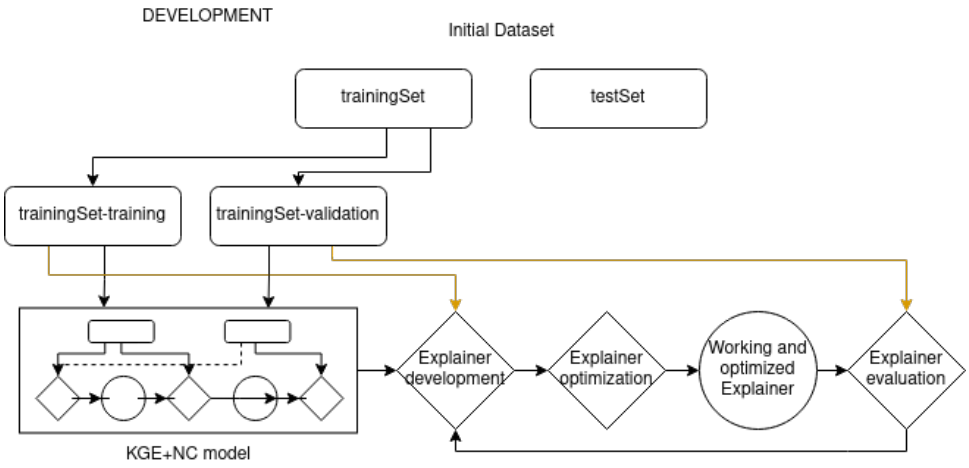

Figure 4.6: Development pipeline for the explainer model.

The development of the explainer models consisted in the general approach that was established to develop and implement the new explanation method. Figure 4.6 shows a summary of the most relevant steps. During the development of the explanation method a small dataset was used, more appropriate for fast iterations and tests, namely the AIFB dataset [52] for which more details are given in the results section. Because the idea was to also use this dataset for the evaluation stage, an extra partition of the dataset was done taking only the original training set and dividing it in training and validation sets. The validation set is used at this stage as a test/holdout set would be used for a final validation of some model, this is because there was the need to test the explainer in unseen instances but without compromising the true test set. An overview of the process is shown in Figure 4.6 and is described as follows:

1. Apply the training pipeline described in Section 4.6.2, but the input dataset is *trainingSet-training* as the *trainingSet* and *trainingSet-validation* as *testSet*;

2. Development of the explainer method where any iterations and debugging procedures were accomplished using instances from the training set;

3. (Optional, for LoFI only) Optimization of the update method parameters implemented in the explainer;

4. The explainer model is obtained; and

5. Evaluation of the explainer model using the effectiveness metrics defined in Section 4.5.1. Also, comparison to random explanations described in Section 4.5.2. The evaluation is accomplished on the instances from the validation set.

6. Steps 2-5 are iterated until the final explanation method is implemented.

The development tries to mimic as much as possible the conditions that will be seen during the final evaluation, namely the fact that the explanation model is designed to explain instances unseen by the prediction model.

4.6.4 Evaluation Pipeline

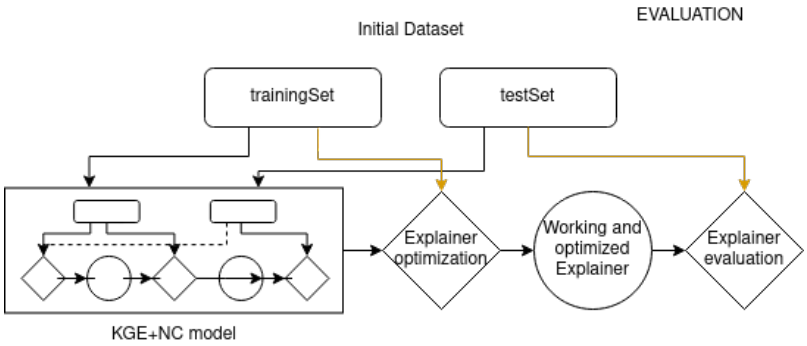

Figure 4.7: Evaluation pipeline for the explainer model.

The evaluation pipeline is the procedure used to validate the explainer using the test/holdout sets and the effectiveness metrics and is illustrated in Figure 4.7. It is described as follows:

1. Apply the training pipeline described in Section 4.6.2, using the complete *trainingSet* and *testSet*;

2. (Optional, for LoFI only) Optimization of the update method parameters implemented in the explainer;

3. The explainer model is obtained; and

4. Evaluation of the explainer model using the effectiveness metrics defined in Section 4.5.1. Also, comparison random explanations described in Section 4.5.2 and GNN solutions described in Section 4.5.3. The evaluation is accomplished on the instances from the test/holdout set.

To apply the explainer in a practical setting, the steps 1-3 described in the pipeline would be performed and subsequently an entity to explain would be given to the model and an explanation would be extracted.

Chapter 5

Results and Discussion

This chapter presents and discusses the results obtained with LoFI and C-KEE. The data used for the validation purposes is introduced in Section 5.1. The performance of the developed explainability solutions is presented in Section 5.2. Section 5.3 presents additional in-depth results for C-KEE alone. Finally, Section 5.4 presents some concrete examples of generated explanations.

5.1 Data

The data is comprised of datasets for Node Classification, primarily based on Knowledge Graphs using RDF. Node Classification with ML over KGs is not the most popular use of graphs but it is of major importance and the focus of this work. In a relatively recent work [26], from major contributors in the field, concerning graph benchmarks, an extensive collection of 14 datasets is published, divided into Node Classification, Link Prediction and graph classification tasks. Despite its large scope, only 2 Knowledge Graphs are proposed and none is for Node Classification. But still, although less prominent then Link Prediction, some benchmarks can be found. This work will consider general purpose benchmark datasets extracted from different knowledge domains.

Originally published in [52] for benchmark purposes, the AIFB, MUTAG and AM datasets are used. From a more recent benchmark [3] the Movie/MDGENRE dataset is used. The first three are also available in the most popular graph frameworks such as PyTorch Geometric (PyG) [16] (as of latest release PyG 2.4.0) or Deep Graph Library (DGL) [84] (as of latest release v1.1.2), although not in RDF graph format.

With some exceptions, the benchmark datasets were chosen for the following characteristics:

- Having variable sizes to enable quick iteration of designs and provide accessible use cases and also to provide varying challenges in computational needs;
- Being found in popular benchmarks; and
- Being about popular culture topics that allow for non-experts to reason about the obtained results and explanations.

The datasets are described as follows:

- The AIFB dataset focuses on the staff, research groups and publications from AIFB, which is the institute of applied informatics of the Karlsruhe Institute of Technology. The dataset is used to predict the affiliation of staff members to one of five research groups. The *affiliation* relation and its inverse (*employs*) must be removed from the dataset because of their direct association to the prediction task. This dataset is used mainly for development purposes and for debugging, due to its very simple nature and very small scale;

- The MUTAG dataset is comprised of complex molecules and the prediction task is to find if a given molecule is potentially carcinogenic or not. The class is given by the *isMutagenic* property which must be removed from the dataset. The highlights of this dataset is that it is larger then AIFB, comes from a more applied use case (biomedical sciences) and is used for binary classification;

- The AM dataset translates a catalog of artifacts from the Amsterdam Museum. In this case, the task is to predict the category of a given artifact such as *Books* or *Paintings*. This dataset is the largest of the original benchmark [52] and should provide a more challenging multi-class classification task than the AIFB;

- The Movie dataset is a subset of Wikidata [71] in the movie domain. The aim is to predict the (single-label) genre of a movie. It's the largest of the datasets listed so far, but its longest path is of length 4, making it a relatively simple use case in terms of graph structure.

Table 5.1: Main statistics for all the benchmark datasets.

Dataset	**Entities**	**Relations**	**Edges**	**Class Labels**				**Facts in 1-Hop for Labeled Entities**	
				Number	Distribution	Total	Test	Mean	Stdev
AIFB	8 285	45	29 043	4	[41, 34, 16, 9]	176	36	19.6	34.7
MUTAG	23 644	23	74 227	2	[62, 38]	340	68	67.9	56.1
AM	881 680	122	5 668 682	11	[35, 13, ..., 1]	1 000	198	11.7	7.78
MDGENRE	349 344	154	1 252 246	12	[55, 14, ..., 1]	8 863	3 005	29.3	21.1

Table 5.1 presents relevant details about the used datasets. The size of the KGs is apparent from the number of entities, relations and edges that compose the graph. The number of classes to be predicted in the Node Classification task and the number of labelled nodes is also shown in the table. Finally, and considering the importance of 1-hop facts in the design of the explainability methods, the table shows statistics of the number of facts present in the labeled entities. Considering that the explainers will only be using a maximum length of 5 facts to extract explanations it may be observed from the start that the pool of facts to choose from is comparatively larger and should present a real challenge.

5.2 Explainers Evaluation

5.2.1 Training of KGE models

For the RDF2Vec models a grid search was performed on the KGE model parameters and the base ML model. The optimization search space was:

- *vector_size* in the range {50, 200, 500};
- *sg* in the range {0, 1};
- *max_depth* in the range {2, 4, 6};
- *max_walks* in the range {200, 500}; and
- Base SL model from {SVC, RandomForestClassifier, GaussianNB}

The results were validated using a Stratified 5-fold cross validation. Appendix A provides additional details about the training results. The best KGE model parameters were obtained and defined independently for each dataset and used in all the trained models that are mentioned in the remainder of this section.

5.2.2 LoFI Explainer Evaluation

For the coupled RDF2Vec and Node Classification (RDF2Vec+NC) model a grid search was performed on the previously selected learning algorithm which was, for all cases, a Random Forest Classifier. The Random Forest was optimized for *max_depth* in the range {2, 4, 6, 8, 10} and the results were validated by using Stratified 5-fold cross validation (details in Appendix A). For each dataset, the final performance measure of the model was taken using the test/holdout set for each dataset.

The explanations are generated considering the class probability condition and a maximum explanation length of 5.

Predictive Model Performance for LoFI

Besides accuracy, F1-score is also the relevant metrics to usually consider in Node Classification tasks, more notably when dealing with imbalanced datasets as is the case with the datasets being used. F1-score is computed using the "weighted" average as it was considered that this would give a balanced account of performance in majority and minority classes, but not as penalizing as using the "macro" average since the goal is to use the metrics to evaluate explanations and the ability to explain a prediction is valued whether that prediction happens in a majority or minority class.

The homologous mimic entities were optimized using a grid search for the following update training parameters: *epochs* in the range {5, 25, 50}, *freeze_embeddings* in the range {True, False} and *init_mimic* in the range {True, False}.

With the homologous mimic entities being such a critical part of the explainability process, it would not be possible to proceed without accessing the behaviour of the predictive models when classifying those entities. In Table 5.2 the performance of the models for the original test entities is compared with the performance for the homologous mimic entities. The results consider an average of 10 models that were trained for each dataset, trained with different seeds to provide a more robust analysis overall. Both accuracy and F1-score are compared and it is observed that the performance of the models tends to decrease slightly. A Wilcoxon Paired-Rank Test with p-value of 0.1 was employed to compare the performance and only in 3 scores the test points to differences with significance.

Table 5.2: RDF2vec+NC models' performance for accuracy and weighted average F1-score, comparing original entities scores with homologous mimics scores.

Dataset	**Trained Models:** RDF2Vec+NC			
	Original Entities		**Homologous Mimics**	
	Accuracy	F1-Score	Accuracy	F1-Score
AIFB	0.87 (0.02)	0.87 (0.02)	0.86 (0.03)	0.86 (0.03)
MUTAG	**0.75 (0.03)**	**0.72 (0.04)**	0.70 (0.05)	0.67 (0.05)
AM	**0.77 (0.04)**	0.76 (0.04)	0.76 (0.04)	0.75 (0.03)
MDGENRE	0.67 (0.01)	0.56 (0.04)	0.66 (0.04)	0.56 (0.01)

In summary, Table 5.2 shows that the model is robust to the "approximated" entities and from another point of view, it can also be said that these results seem to indicate that the homologous mimics may be quite good at replicating the original entities.

Test Results for Sufficient Explanations

This section describes the results obtained for all datasets when searching for the counterfactual sufficient explanations with LoFI.

It was decided early on that to validate the LoFI solution on all the test samples for the larger datasets (AM and MDGENRE), would be unreasonable, due to the amount of time that the evaluation procedure takes, having to completely train everything from scratch to validate each explanation individually. For AM and MDGENRE, a stratified sample of 50 instances from each dataset was randomly sampled and used in all the subsequent test results reported for LoFI. For AIFB and MUTAG the complete test/holdout set was used.

The results are evaluated using the ΔAccuracy and ΔF1-Score scores. Recall that these scores are the difference between the scores of the original model and the model completely trained from scratch but with the difference of removing or adding the explaining facts to the initial KG. For a sufficient explanation the goal is that the accuracy or F1-score of the ablated entity increases the most such that the ΔAccuracy and ΔF1-Score have the least negative value possible or even a positive value. The analysis in focused on the F1-scores, accuracy results can be found in Appendix B considering that the results are very similar.

The evaluation compares the results obtained with LoFI with the results obtained by considering random sufficient explanations. In order to provide a more truthful comparison there is some information from the LoFI explanations that is carried to the random explanations and that is the quantity of facts used. For example, if for some entity the explanation in LoFI was composed of only one fact then only one random fact was used in the Random variant, if for some entity the explanation in LoFI was composed of three facts then three facts were used in the Random variant. In principle, this should allow for a more direct comparison and if some entity was harder to explain with LoFI, then the equivalent number of facts is considered for Random.

A statistical analysis for significance is performed for the 10 models per dataset, trained with different seeds, and where for each model one set of LoFI explanations and one set of Random explanations are extracted for the test instances.

Previous to the test for significance, normality test appropriate for small samples was employed and showed that some samples were significantly different from a normal distribution, according to the Shapiro-Wilk Test for Normality. Because of this the following statistics tests are nonparametric. Also, it is worth mentioning that the purpose of this analysis was always to be able to test for significance and compare the samples, not to derive information for the population represented in the samples, and for that case the nonparametric tests are sufficiently appropriate.

For the nonparametric tests, the Related-Samples Wilcoxon Signed Rank Test was employed to test for differences between LoFI and Random explanations with:

- Null Hypothesis: the median of differences between Random and LoFI scores equals 0.

The significance level chosen for the tests is $\alpha = 0.1$. Since there is no previous significance level that could be referred to in similar situations and the experience with this type of samples being evaluated was non-existing prior to this work, this level was considered as appropriate for the occasion.

Table 5.3: Wilcoxon signed rank statistical test results for the test set for all datasets, for weighted average F1-scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	**Sig.**	**Decision**
AIFB	0.005	**Reject the null hypothesis**
MUTAG	0.308	Retain the null hypothesis
AM	0.013	**Reject the null hypothesis**
MDGENRE	0.005	**Reject the null hypothesis**

Table 5.3 shows the decisions obtained from the tests performed on the F1-scores. For three datasets the bold highlight means that the null hypothesis is rejected, and because there is a positive difference between the samples this supports the hypothesis that the sufficient explanations obtained with LoFI are indeed performing significantly better than Random.

Figure 5.1 shows the plots for the actual F1-scores and gives a clearer notion of what is happening with the sufficient scores. The decisions from the previous tests are illustrated in the plot by a red outline of the corresponding boxplot where the differences were indeed significant between LoFI and Random for each dataset. With the exception of MUTAG, where the scores are very similar to random explanations, in the other datasets it could be said that there is a clear distinction between LoFI explanations and random explanations in favour of the LoFI explanations.

In general the final results obtained for sufficient explanations are very satisfactory and they show that it's possible to extract relevant explanations with the proposed explainability method.

Test Results for Necessary Explanations

This section describes the results obtained for all datasets when searching for the counterfactual necessary explanations.

For a necessary explanation the goal is that the accuracy or F1-score of the ablated entity decreases the most such that the ΔAccuracy and ΔF1-Score have the most negative value possible.

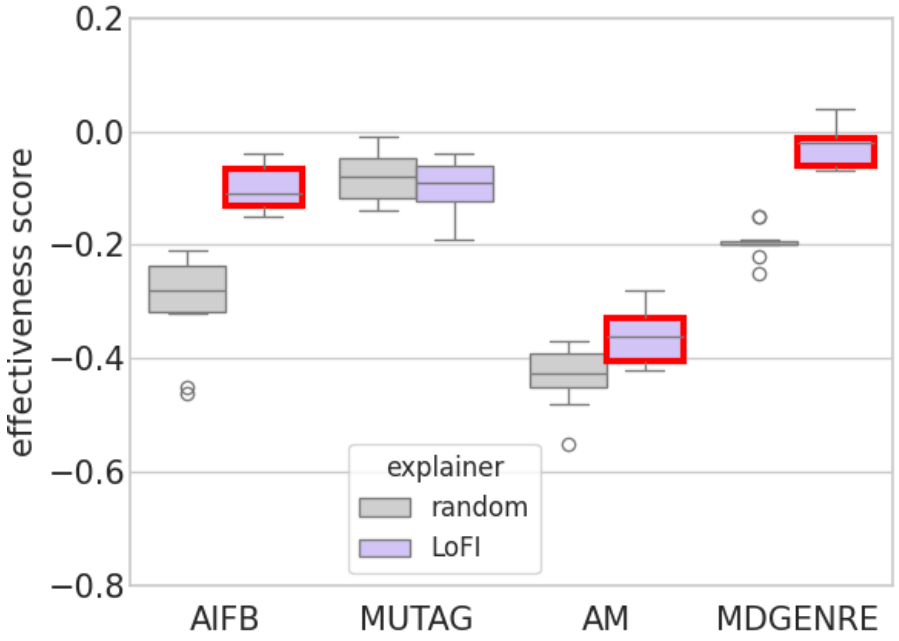

Figure 5.1: Results for the test set for all datasets, for weighted average F1-scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

The evaluation again compares the results obtained with LoFI with the results obtained by considering random sufficient explanations. For a more truthful comparison, the same number of facts removed by LoFI explanations is used to extract random necessary explanations.

The statistical analysis for the necessary explanations was accomplished in similar fashion to the sufficient counterpart in the previous section.

Again, normality tests showed that some samples were significantly different from the normal distribution and the same nonparametric tests were employed just like in the statistical analysis of the necessary test results.

Table 5.4: Wilcoxon signed rank statistical test results for the test set for all datasets, for weighted average F1-scores, for necessary explanations, for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	**Sig.**	**Decision**
AIFB	0.007	Reject the null hypothesis
MUTAG	0.314	Retain the null hypothesis
AM	0.333	Retain the null hypothesis
MDGENRE	0.634	Retain the null hypothesis

Table 5.4 shows the decisions obtained from the tests performed on the F1-scores. A result that would support the case for the LoFI model to be significantly better than random would be to have a decision to reject the null hypothesis and to have negative differences between the LoFI scores and Random scores (the LoFI scores would be more negative than the Random scores) and this case would be highlighted in bold in the table.

Table 5.4 shows that for three datasets the results support the null hypothesis, so there are no observ-

able differences between LoFI and Random according to these tests. For AIFB the results do no support the null hypothesis but the differences between LoFI and Random are positive which means that, in the context of these explanations, random explanations actually would perform better than LoFI at finding relevant explanations, according to these statistical tests.

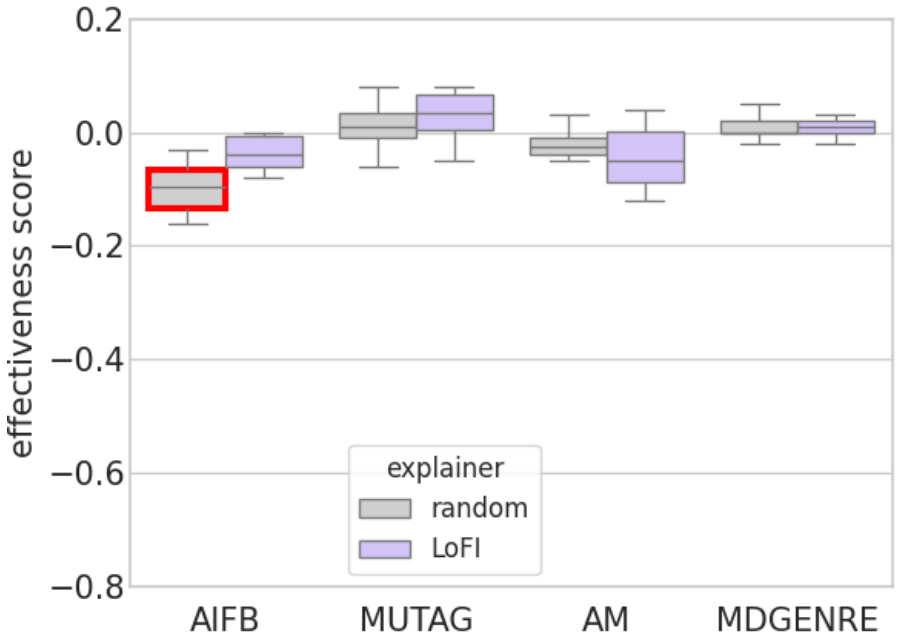

Figure 5.2: Results for the test set for all datasets, for weighted average F1-scores, for necessary explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

Figure 5.2 shows the plots for the actual scores used to carry out the statistical tests. The boxplots show that AIFB explanations are clearly worse with LoFI, slighly worse in MUTAG, but slightly better than random with AM and MDGENRE.

The less successful results may be due to the strong variance existing in the embeddings representations obtained with RDF2Vec and Word2vec solutions which are considered quite unstable methods, and by using an approximation method to represent the entity (as is the case with the update method) it's possible that to remove only a few facts is not enough for the update method to truthfully capture what would happen with the original entity for a brand new model trained from scratch. Also, by just removing one or two facts the impact in the relevance measure may be small in the majority of cases and a selected fact may be only selected based on a small difference that when tested does not have any meaning.

5.2.3 C-KEE Explainer Evaluation

The explanations are generated considering the class probability condition and a maximum explanation length of 5.

Table 5.5: RDF2vec+NC models' performance for accuracy and weighted average F1-scores, comparing the original model using original entities scores with the global aggregate model using global aggregate entities scores.

Dataset	**Original RDF2Vec+NC with Original Entities**		**Global Aggr. RDF2Vec+NC with Aggregate Entities**	
	Accuracy	F1-Score	Accuracy	F1-Score
AIFB	0.88 (0.04)	0.88 (0.04)	0.87 (0.03)	0.87 (0.03)
MUTAG	0.72 (0.04)	0.66 (0.07)	0.71 (0.03)	0.68 (0.04)
AM	0.82 (0.03)	0.80 (0.03)	**0.85 (0.01)**	**0.84 (0.01)**
MDGENRE	**0.64 (0.01)**	**0.53 (0.01)**	0.63 (0.01)	0.53 (0.01)

Predictive Model Performance for C-KEE

With the aggregate entities representation being such a critical part of the explainability process, Table 5.5 shows the performance of the models for the original test entities compared with the performance for the aggregate test entities. Both accuracy and F1-scores are compared and it can be observed that the performance of the models is very similar. Using the Wilcoxon Paired-Rank Test with p-value of 0.1 it can be observed that the original entities model is significantly better in the MDGENRE dataset and the aggregate entities model is significantly better in the AM dataset.

In summary, Table 5.5 shows that the aggregate model has an equivalent performance when compared to the original model.

Test Results for Sufficient Explanations

This section describes the results obtained for all datasets when searching for the counterfactual sufficient explanations using the C-KEE explainer.

The evaluation procedure, evaluation metrics, statistical analysis and random explanations approach are in all similar to the ones used in the LoFI explainer evaluation. The evaluation procedure for C-KEE does not require re-training the model from scratch for each test instance, making it notably faster. For this reason, the results for the AM and MDGENRE are obtained considering the complete test set, contrary to what happened with LoFI.

Table 5.6: Wilcoxon signed rank statistical test results for the test set for all datasets, for weighted average F1-scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	**Sig.**	**Decision**
AIFB	0.005	**Reject the null hypothesis**
MUTAG	0.114	Retain the null hypothesis
AM	0.005	**Reject the null hypothesis**
MDGENRE	0.005	**Reject the null hypothesis**

Table 5.6 shows the decisions obtained from the tests performed on the F1-scores. For three datasets the null hypothesis is rejected and there is a positive difference between the samples which supports the hypothesis that the sufficient explanations obtained with C-KEE are performing significantly better than Random.

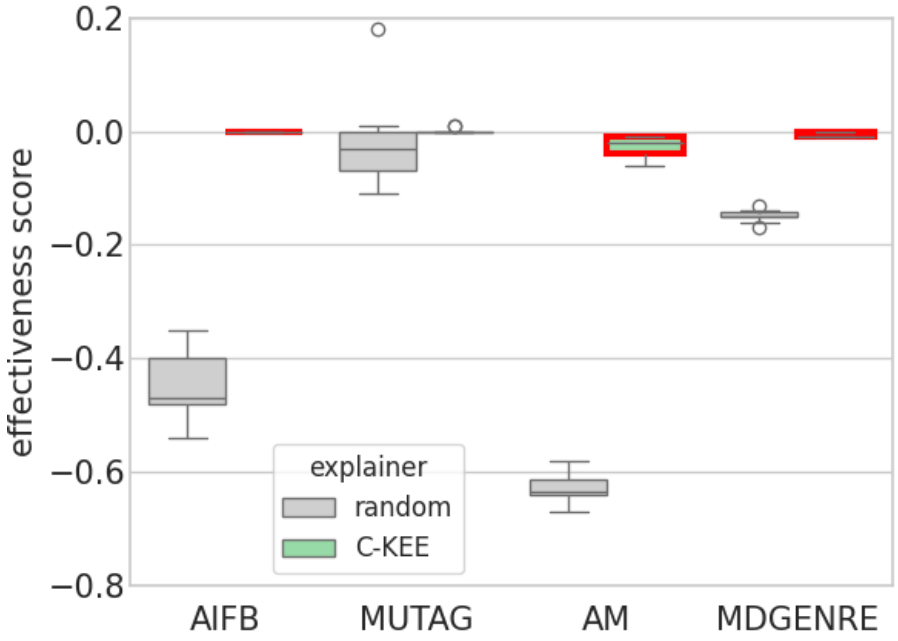

Figure 5.3: Results for the test set for all datasets, for weighted average F1-scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

Figure 5.3 shows the plots for the actual F1-scores. C-KEE performs extremely well in all the datasets showing values close to or equal to zero difference between the entities defined using only the found sufficient facts and the original entities. Such is the case also for MUTAG, were no statistical significance was found due to the fact that the random explanations also perform well.

Test Results for Necessary Explanations

This section describes the results obtained for all datasets when searching for the counterfactual necessary explanations using the C-KEE explainer.

Table 5.7: Wilcoxon signed rank statistical test results for the test set for all datasets, for weighted average F1-scores, for necessary explanations, for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	Sig.	Decision
AIFB	0.003	**Reject the null hypothesis**
MUTAG	0.155	Retain the null hypothesis
AM	0.003	**Retain the null hypothesis**
MDGENRE	0.003	**Retain the null hypothesis**

Table 5.7 shows the decisions obtained from the tests performed on the F1-scores. For the three dataset in bold the null hypothesis is rejected and there is a negative difference between the samples which supports the hypothesis that the necessary explanations obtained with C-KEE are performing significantly better than Random.

Figure 5.4 shows the plots for the actual scores used to carry out the statistical tests. The boxplots

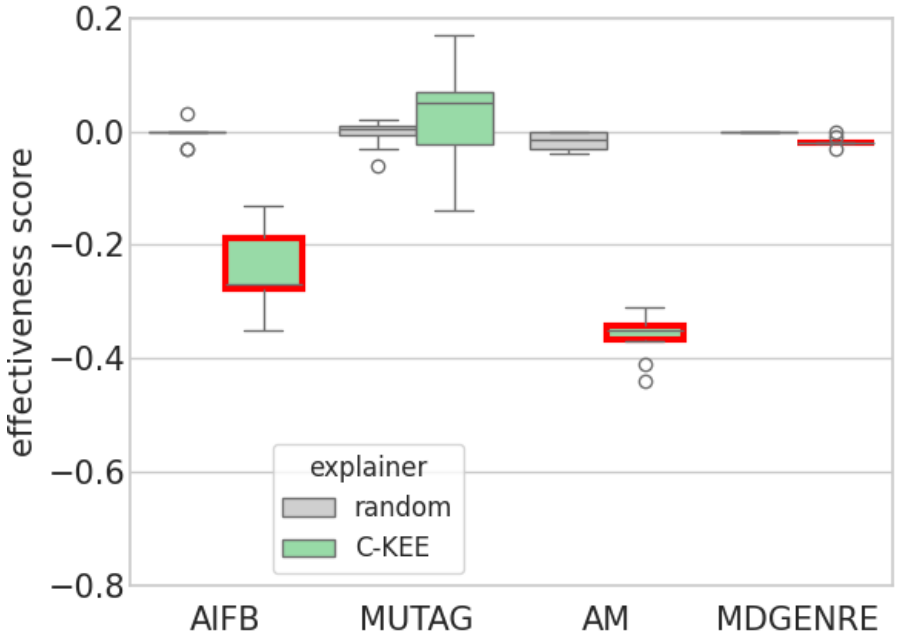

Figure 5.4: Results for the test set for all datasets, for weighted average F1-scores, for necessary explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

show that the explanations for AIFB, AM and MDGENRE are notably better then Random. Only for MUTAG does the model struggle to find necessary explanations.

5.2.4 Comparison of LoFI, C-KEE and GNN Solutions

This section presents a brief comparison of LoFI and C-KEE against each other and, were possible, against alternative GNN models and their respective explainer solutions. The obtained explanations for LoFI and C-KEE were generated with the same explanation conditions, both using the same maximum length per explanations and the class probability condition, and as such are fully comparable.

Comparison of Sufficient Explanations

For GNN models, explanations might be found n-hops away from the entity they are explaining, and the concept of keeping only the sufficient explanations does not translate well for such cases. For this reason, for sufficient explanations, LoFI and C-KEE are compared only with each other.

Table 5.8: Mann-Whitney U statistical test results for the test set for all datasets, for weighted average F1-scores, for sufficient explanations, for maximum explanation length of 5 - Node Classification with KGE explained with C-KEE and LoFI, with 10 independent runs for each model.

	AIFB	**MUTAG**	**AM**	**MDGENRE**
C-KEE-LoFI	**<0.001**	**<0.001**	**<0.001**	**0.023**

Table 5.8 shows the results of an Independent-Samples Mann-Whitney U test considering a p-value of 0.10. Results in bold show the rejection of the null hypothesis and thus point to a significant difference

between distributions. For all datasets the C-KEE mean ranks are higher then LoFI meaning that C-KEE performs better across the range for sufficient explanations.

Comparison of Necessary Explanations

Table 5.9: Mean and standard deviation of accuracy and weighted average F1-score for the performance of the GCN trained classification models.

Dataset	Accuracy	F1-Score
AIFB	0.87 (0.04)	0.86 (0.05)
MUTAG	0.72 (0.03)	0.70 (0.04)
AM	0.72 (0.05)	0.70 (0.04)
MDGENRE	0.64 (0.00)	0.54 (0.01)

For the GCN [32] model, an architecture with 2 convolutional layers was used, the size of the output of layer 1/input of layer 2 is 16, between layers there is ReLU and dropout, the prediction step is accomplished with a softmax followed by a logarithm. Again, for each dataset, the final performance measure of the model was taken using the test/holdout set for each dataset. As before, 10 models were trained for each dataset, taking the mean and standard deviation for the accuracy and F1-scores as shown in Table 5.9.

The goal is to compare end-to-end solutions that rely on KGE against solutions that rely on GNNs since these are the closest alternatives present in the literature. The GNN solution uses GNN prediction models used in conjunction with explainability solutions for GNNs to output an explanation that could be interpreted as a necessary explanation and compared to LoFI and C-KEE.

For such comparison, a statistical analysis for significance is again performed. So, for each of the 10 randomly seeded and trained GNN models for each dataset, explanations are extracted using two different GNN explainers, the GradExplainer [60] and the GNNExplainer [80]. The models for LoFI and C-KEE are the same used in the previous sections because in this case the samples between each explainer solution are completely independent.

The statistics are again nonparametric. Because the samples are independent from each other, the Independent Samples Mann-Whitney U Test was employed to test for differences between the different solutions. The test is the following:

- Null Hypothesis: the distribution of scores is the same between Explainer 1 and Explainer 2.

The significance level chosen for the tests is again $\alpha = 0.1$.

Table 5.10 shows the decisions obtained from the tests for the F1-scores. Results in bold or underline show the rejection of the null hypothesis and thus point to a significant difference between distributions. Bold highlights the cases were Explainer 1 is better and underline highlights the cases were Explainer 2 is better. The results show that, in general, LoFI and C-KEE output better necessary explanations than the GNN explainers for the AIFB and AM datasets. There is only one instance were one GNN explainer is significantly better.

Figure 5.5 shows that, overall, C-KEE performs much better than LoFI in most datasets, and much better than the GNN explainers in AIFB and AM and is in no case worse than those.

Table 5.10: Mann-Whitney U statistical test results for the test set for all datasets, for weighted average F1-scores, for necessary explanations, for maximum explanation length of 5 - Node Classification with KGE explained with C-KEE and LoFI, Node Classification with GNN explained with GNN explainer models, with 10 independent runs for each model.

	AIFB	**MUTAG**	**AM**	**MDGENRE**
LoFI-GradExplainer	**0.004**	0.280	**0.007**	0.143
LoFI-GNNExplainer	**0.011**	0.247	0.436	0.023
C-KEE-GradExplainer	**<0.001**	0.280	**<0.001**	0.393
C-KEE-GNNExplainer	**<0.001**	0.280	**<0.001**	0.436
C-KEE-LoFI	**<0.001**	0.739	**<0.001**	**0.002**

5.2.5 Performance

All the experiments with Knowledge Graph Embeddings, LoFI and C-KEE ran on a server using at most 24 CPUs Intel Core(TM) i9-12900KF with max 5.2 GHz and 126 GB RAM.

For the performance aspects of the developed explanations, the critical measurement is the computation time required to generate an explanation. Regarding the explanation time per instance, Figure 5.6 shows that C-KEE is several orders of magnitude faster than LoFI, with C-KEE having median explanation times between 0.141 and 0.784 seconds and LoFI between 16.3 and 648 seconds.

For LoFI it can be observed that the explanation time grows with the size of the dataset. On the contrary, C-KEE explanations times seem to be independent of dataset size, which is a very valuable attribute for scalability. C-KEE explanation times seem to be correlated to the number of 1-hop facts for the entities in the dataset, but since most datasets usually scale much more in terms of number of entities than in terms of links for each entity this feature should not cause any limitation in practical use.

To speed up the total running time several parts of the code were parallelized, most notably the explanation algorithm was parallelized because a considerable part of the prediction pipeline originally ran on a single CPU, and the evaluation algorithm was parallelized. Also, in critical parts of the algorithm, the datasets were cached since the loading time for the original loading procedure of RDF graphs was also a critical component influencing computation times. Despite this, the evaluation time for LoFI was a critical component since for evaluating the necessary and sufficient explanations for a single model it took between 4.8 minutes of run time in AIFB (to evaluate all the test instances) and 3.1 hours in MDGENRE (to evaluate the 50 sampled test instances). It's unusual that evaluation times should make up such a large part of the process but due to the nature of the problem this was the best found solution. On the other hand, the evaluation time for C-KEE was completely negligible in comparison.

5.2.6 General Discussion of LoFI and C-KEE Results

Across the results, the MUTAG dataset proved consistently difficult to generate satisfactory evaluation results for. Given that this dataset has a number of facts in 1-hop neighbourhood higher than the others, this fact could have affected the ability of the explainer model to find relevant explanations. Also, for a binary classification dataset, the performance of the predictive models was far from perfect, and considering the approach of the explainer models, particularly in the necessary explanations, in this dataset it would be easier to end up explaining incorrectly classified instances and while doing so inadvertently end up changing an incorrect prediction to a correct prediction. This effect may have contributed to poor

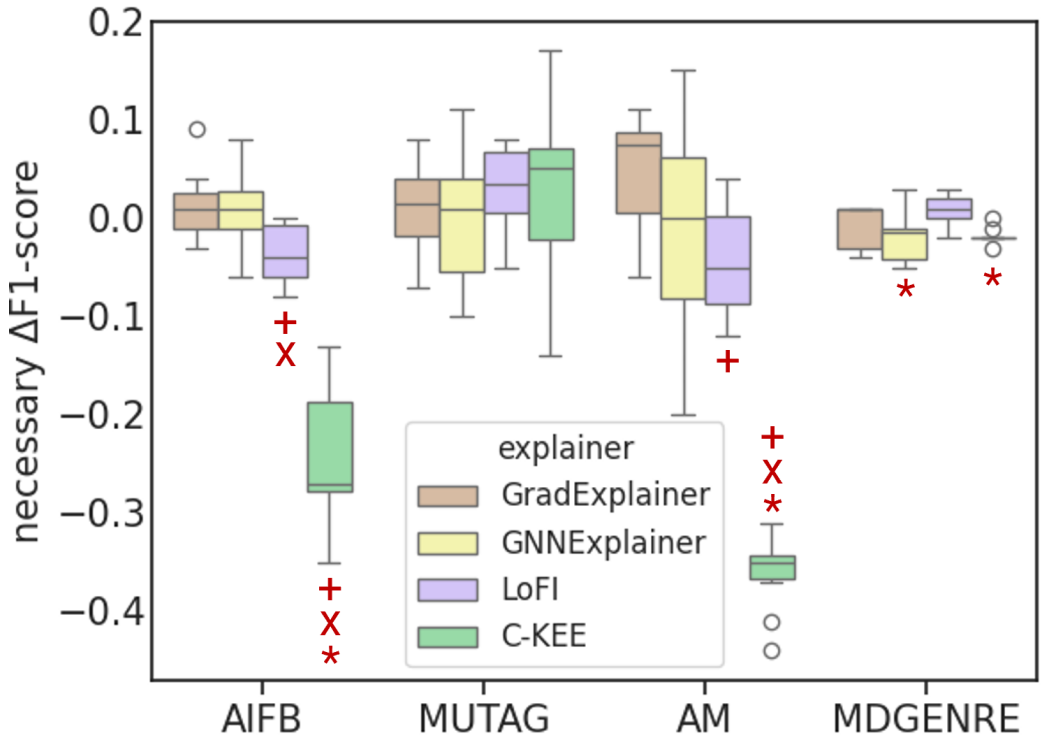

Figure 5.5: Results for the test set for all datasets, for weighted average F1-scores, for necessary explanations, for maximum explanation length of 5 - Node Classification with KGE explained with C-KEE and LoFI, Node Classification with GNN explained with GNN explainer models, with 10 independent runs for each model. Statistically significant results, per dataset, highlighted by the red marks: "**+**" better than GradExplainer, "**X**" better than GNNExplainer, "*" better than LoFI.

.

results considering that the evaluation metric for explainability is based on the accuracy and F1-scores of the models.

In general the final results obtained for sufficient explanations are very satisfactory with both LoFI and C-KEE and show that it's possible to extract relevant explanations with the proposed explainability method. The successful results obtained with this type of explanation are even more relevant because sufficient explanations can be considered as synthesized representations of an entity for some prediction. Because of what the sufficiency concept implies, they should be very easy to reason with and easily understood from a human user standpoint in practical applications.

For necessary explanations C-KEE proved much more performing than LoFI. Taking these results along with the much faster explanation times using C-KEE, at this stage it seems that C-KEE is the better solution for the problem at hand.

5.3 Additional Results for the C-KEE Explainer

This section presents additional results for the C-KEE explainability method. This additional work was accomplished after all the evaluation procedures described in the previous sections were done and observing that C-KEE showed the most promise for further exploration.

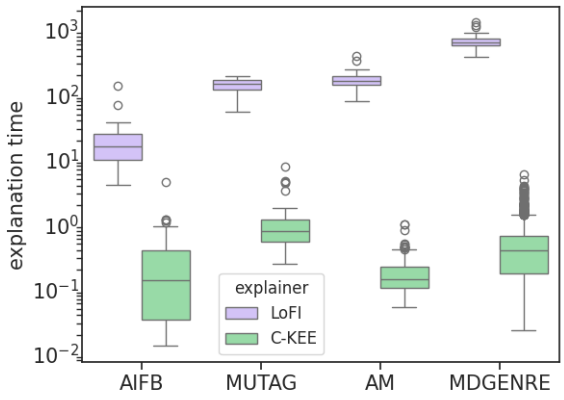

Figure 5.6: Comparison of explanation times (in seconds) between C-KEE and LoFI, for all datasets.

For this new set of results, and with the explainability method already working, a new evaluation method was defined, more aligned with the current trends of publications in the research area associated with Node Classification with KGE. These experiments ran using 10-fold cross-validation on the complete dataset for each of the previous datasets. In this case there was no fine-tuning of KGE models as it is prohibitive to employ a grid search approach for KGE in each fold. Five different KGE models were evaluated: RDF2Vec [53], ComplEx [69], distMult [77], TransH [75] and TransE [5]. Three different types of ML models were used: RandomForest [6], XGBoost [10] and Artificial Neural Networks [35]. In each fold it was employed a nested grid-search optimization for the ML models. The parameters for KGE and ML models are available in Appendix D). Explanations were independently generated and evaluated for two different conditions, the class probability condition and the class change condition. Explanations were extracted for both a predefined maximum explanations length of 5, as before, but also for a maximum length of 1 (single fact explanations).

The results for a subset of the experiments are shown in this section in Tables 5.11, 5.12 and 5.13. The remainder of the results can be found in Appendix E.

Predictive Model Performance

Table 5.11 shows the comparison of predictive performance between the original models and the C-KEE models for RDF2Vec, ComplEx and distMult methods coupled with a RandomForest classifier. Statistical significance is investigated using Related-Samples Wilcoxon Signed Rank Test with a p-value of 0.05 (a stricter significance value than before) and highlighted in bold. The results show that C-KEE global representation model performs comparably or better than the original models (baseline) in nearly all cases. C-KEE is particularly effective in the AM dataset where it always shows significantly better results. For the worse results obtained with the original models in the MDGENRE dataset, C-KEE is also able to significantly improve the results.

5.3.1 Explainer Model Performance

Table 5.12 shows the results for the performance of explanations generated with the class probability condition. Statistical significance is again evaluated with the Related-Samples Wilcoxon Signed Rank

Table 5.11: Mean and standard deviation of precision, recall, weighted average F1-score and accuracy (Pr, Re, F1, Ac) comparing C-KEE global approach to the baseline when coupled with different KGE methods and the Random Forest classifier. Highlighted results (bold) are statistically significantly better than the direct comparison.

		AIFB		**MUTAG**		**AM**		**MDGENRE**	
		baseline	C-KEE	baseline	C-KEE	baseline	C-KEE	baseline	C-KEE
RDF2Vec	Pr	0.941 (0.028)	0.951 (0.033)	0.765 (0.118)	0.728 (0.135)	0.677 (0.09)	**0.781 (0.05)**	0.545 (0.071)	0.524 (0.071)
	Re	0.926 (0.063)	0.937 (0.074)	0.744 (0.121)	0.7 (0.109)	0.679 (0.046)	**0.765 (0.063)**	0.632 (0.01)	0.625 (0.009)
	F1	0.923 (0.052)	0.935 (0.061)	0.73 (0.133)	0.679 (0.129)	0.639 (0.062)	**0.757 (0.053)**	**0.532 (0.014)**	0.519 (0.016)
	Ac	0.926 (0.047)	0.937 (0.057)	0.744 (0.114)	0.7 (0.118)	0.679 (0.041)	**0.765 (0.037)**	0.632 (0.02)	0.625 (0.021)
ComplEx	Pr	0.768 (0.151)	**0.868 (0.142)**	0.74 (0.08)	0.721 (0.102)	0.451 (0.068)	**0.685 (0.057)**	0.439 (0.015)	0.463 (0.04)
	Re	0.79 (0.117)	0.858 (0.138)	0.724 (0.063)	0.688 (0.063)	0.506 (0.046)	**0.682 (0.04)**	0.61 (0.002)	**0.62 (0.004)**
	F1	0.757 (0.135)	**0.839 (0.146)**	0.708 (0.072)	0.652 (0.086)	0.429 (0.052)	**0.646 (0.044)**	0.494 (0.003)	**0.507 (0.004)**
	Ac	0.79 (0.08)	0.858 (0.089)	0.724 (0.067)	0.688 (0.07)	0.506 (0.06)	**0.682 (0.037)**	0.61 (0.02)	**0.62 (0.018)**
distMult	Pr	0.923 (0.029)	0.917 (0.098)	0.758 (0.089)	0.705 (0.081)	0.521 (0.116)	**0.757 (0.111)**	0.471 (0.033)	0.491 (0.049)
	Re	0.903 (0.096)	0.897 (0.133)	**0.721 (0.078)**	0.682 (0.097)	0.572 (0.09)	**0.75 (0.061)**	0.629 (0.004)	**0.638 (0.003)**
	F1	0.899 (0.072)	0.894 (0.122)	0.696 (0.095)	0.651 (0.115)	0.507 (0.089)	**0.729 (0.069)**	0.515 (0.004)	**0.523 (0.004)**
	Ac	0.903 (0.054)	0.897 (0.085)	**0.721 (0.088)**	0.682 (0.106)	0.572 (0.06)	**0.75 (0.04)**	0.629 (0.018)	**0.638 (0.019)**

Test with a p-value of 0.05. In the case of necessary explanations, the desired behaviour is that the performance based on C-KEE explanations is significantly lower than the performance of the original models and the performance of models based on random explanations, and these results are highlighted in bold and italics, respectively. For sufficient explanations, the desired behaviour is that the predictive performance is not significantly lower when compared to the original models (highlighted in bold, if true) but is significantly better when compared to random explanations (highlighted in italics, if true).

The results show that, for necessary explanations, when a single triple is used as an explanation, the effectiveness of explanations is significant for nearly all metrics in the AIFB and AM, whereas for MUTAG this is only achieved with RDF2Vec. When a maximum of 5 triples where employed (compound explanation), there were relevant improvements in the cases where a single explanation had already produced significant results but there was not much improvement in the overall quantity of significant results.

The results shows that, for sufficient explanations, When a single triple is used as an explanation, the performance difference from the C-KEE to the original model is not significantly lower for all metrics and KGE methods in the AIFB and MUTAG datasets, and for distMult, TransH and TransE in MDGENRE. When 5 triples where employed, the main improvement was the achievement of significant results in the MDGENRE dataset.

Table 5.13 shows the results for the performance of explanations generated with the class change condition. Interestingly, when up to five triples can be used in a necessary explanation, C-KEE is able to achieve significantly effective explanations for all metrics in AIFB and AM regardless of the KGE method, for MUTAG with RDF2Vec, ComplEx and distMult and for MDGENRE with RDF2Vec, ComplEx, TransH and TransE. Analysing the results from the KGE methods perspective, C-KEE is able to achieve significantly effective explanations with RDF2Vec and ComplEx for all datasets and metrics and nearly all metrics for the other KGE methods (with the exception of TransE on MUTAG).

For sufficient explanations, C-KEE is always significantly better than random explanations across all datasets and metrics. When up to 5 triples are employed, it was possible to achieve significantly improved results on all datasets and metrics, both comparing to the original models performance and

Table 5.12: Mean explanation effectiveness based on the class probability condition with maximum length of 1 (simple) and 5 (comp), based on the precision (Pr), recall (Re), weighted average F1-score (F1) and accuracy (Ac) variation using RandomForest. $rand_s$ corresponds to performance using randomly generated explanations of length 1, and $rand_c$ of up to 5. Results for TransE and TransH are in the Appendix E. Scores are in bold when they represent a significant improvement over the global approach and in italics for random explanations.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$
Necessary																	
RDF2Vec	ΔPr	***-0.15***	0.0	***-0.158***	0.0	**-0.107**	-0.014	**-0.114**	-0.025	***-0.287***	-0.006	***-0.333***	-0.001	-0.026	-0.011	***-0.045***	0.0
	ΔRe	***-0.246***	0.0	***-0.263***	0.0	**-0.091**	-0.009	***-0.103***	-0.018	***-0.281***	-0.007	***-0.328***	-0.009	***-0.009***	0.001	***-0.025***	0.002
	ΔF1	***-0.253***	0.0	***-0.272***	0.0	-0.079	-0.01	-0.083	-0.022	***-0.293***	-0.005	***-0.343***	-0.007	*-0.008*	0.001	***-0.016***	0.002
	ΔAc	***-0.246***	0.0	***-0.263***	0.0	**-0.091**	-0.009	***-0.103***	-0.018	***-0.281***	-0.007	***-0.328***	-0.009	***-0.009***	0.001	***-0.025***	0.002
ComplEx	ΔPr	***-0.205***	0.001	***-0.352***	0.0	-0.046	0.01	-0.049	-0.021	***-0.223***	-0.021	***-0.358***	-0.03	-0.01	0.01	-0.039	-0.013
	ΔRe	***-0.269***	-0.0	***-0.422***	0.005	-0.029	-0.003	-0.038	-0.009	***-0.176***	-0.02	***-0.281***	-0.024	***-0.024***	-0.002	***-0.047***	-0.001
	ΔF1	***-0.28***	0.001	***-0.446***	0.006	-0.002	-0.007	-0.008	-0.007	***-0.207***	-0.021	***-0.324***	-0.028	***-0.022***	-0.002	***-0.041***	-0.001
	ΔAc	***-0.269***	-0.0	***-0.422***	0.005	-0.029	-0.003	-0.038	-0.009	***-0.176***	-0.02	***-0.281***	-0.024	***-0.024***	-0.002	***-0.047***	-0.001
distMult	ΔPr	***-0.162***	0.003	***-0.298***	-0.006	-0.003	0.034	-0.016	0.029	***-0.252***	-0.017	***-0.381***	-0.035	-0.005	0.003	-0.03	-0.022
	ΔRe	***-0.228***	0.0	***-0.341***	-0.0	0.003	0.021	-0.009	0.018	***-0.195***	-0.014	***-0.324***	-0.025	***-0.013***	0.001	***-0.034***	-0.0
	ΔF1	***-0.236***	0.003	***-0.369***	-0.001	0.021	0.025	0.013	0.016	***-0.231***	-0.015	***-0.362***	-0.027	***-0.01***	0.001	***-0.028***	-0.001
	ΔAc	***-0.228***	0.0	***-0.341***	-0.0	0.003	0.021	-0.009	0.018	***-0.195***	-0.014	***-0.324***	-0.025	***-0.013***	0.001	***-0.034***	-0.0
Sufficient																	
RDF2Vec	ΔPr	***0.0***	-0.454	***0.0***	-0.349	***0.0***	-0.221	***0.0***	-0.228	*-0.033*	-0.525	*-0.032*	-0.542	*-0.04*	-0.159	***-0.01***	-0.159
	ΔRe	***0.0***	-0.466	***0.0***	-0.393	***0.0***	-0.188	***0.0***	-0.206	*-0.034*	-0.49	*-0.032*	-0.545	*-0.002*	-0.253	***-0.001***	-0.227
	ΔF1	***0.0***	-0.482	***0.0***	-0.401	***0.0***	-0.179	***0.0***	-0.191	*-0.04*	-0.547	*-0.038*	-0.57	*-0.006*	-0.174	*-0.004*	-0.155
	ΔAc	***0.0***	-0.466	***0.0***	-0.393	***0.0***	-0.188	***0.0***	-0.206	*-0.034*	-0.49	*-0.032*	-0.545	*-0.002*	-0.253	***-0.001***	-0.227
ComplEx	ΔPr	***0.0***	-0.635	***0.0***	-0.544	***0.0***	-0.173	***0.0***	-0.196	*-0.219*	-0.552	*-0.1*	-0.539	*-0.03*	-0.124	***-0.022***	-0.121
	ΔRe	***0.0***	-0.587	***0.0***	-0.559	***0.0***	-0.182	***0.0***	-0.206	*-0.118*	-0.431	*-0.055*	-0.413	***-0.001***	-0.175	***-0.001***	-0.167
	ΔF1	***0.0***	-0.621	***0.0***	-0.588	***0.0***	-0.144	***0.0***	-0.169	*-0.17*	-0.484	*-0.079*	-0.473	*-0.002*	-0.125	***-0.002***	-0.12
	ΔAc	***0.0***	-0.587	***0.0***	-0.559	***0.0***	-0.182	***0.0***	-0.206	*-0.118*	-0.431	*-0.055*	-0.413	***-0.001***	-0.175	***-0.001***	-0.167
distMult	ΔPr	***-0.031***	-0.555	***-0.031***	-0.607	***0.0***	-0.161	***0.0***	-0.133	*-0.178*	-0.56	*-0.06*	-0.557	***-0.021***	-0.088	***-0.021***	-0.07
	ΔRe	***-0.017***	-0.551	***-0.017***	-0.597	***0.0***	-0.153	***0.0***	-0.118	*-0.111*	-0.428	*-0.04*	-0.428	***0.0***	-0.137	***0.0***	-0.111
	ΔF1	***-0.025***	-0.58	***-0.025***	-0.622	***0.0***	-0.125	**0.0**	-0.094	*-0.153*	-0.522	*-0.049*	-0.525	***-0.001***	-0.1	***-0.001***	-0.083
	ΔAc	***-0.017***	-0.551	***-0.017***	-0.597	***0.0***	-0.153	***0.0***	-0.118	*-0.111*	-0.428	*-0.04*	-0.428	***0.0***	-0.137	***0.0***	-0.111

random explanations performance.

When comparing the class change condition with the class probability condition results, it can be observed that when using 5 triples many of the class change results are overwhelmingly better, for example with RDF2Vec on AIFB, MUTAG and MDGENRE the scores mores than double, and in many other cases the results improve considerably.

Overall, C-KEE is able to achieve significant effectiveness of explanations for all methods, datasets and metrics for both the sufficient and necessary scenarios (except for TransE and TransH w.r.t. precision on MUTAG), and the use of compound explanations and class change condition presented considerable improvements compared to single explanations and the class probability condition.

It is worth noting that, as shown in Figure 5.7, when using the class change condition, C-KEE is able to generate sufficient explanations that meet the class change condition for nearly all nodes in all datasets and KGE methods using a maximum explanation length of five. However, for necessary explanations this coverage is reduced. Nevertheless, C-KEE outputs explanations for all nodes using the most relevant explanation criterium. The figure also confirms that the addition of compound explanations has an important contribution on the rate of successful necessary explanations found, where it leads to relevant improvements in AIFB and AM and more than doubles the successful explanations in MUTAG

Table 5.13: Mean explanation effectiveness based on the class change condition with maximum length of 1 (simple) and 5 (comp), based on the precision (Pr), recall (Re), weighted average F1-score (F1) and accuracy (Ac) variation using RandomForest. $rand_s$ corresponds to performance using randomly generated explanations of length 1, and $rand_c$ of up to 5. Results for TransE and TransH are in the Appendix E. Scores are in bold when they represent a significant improvement over the global approach and in italics for random explanations.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$
Necessary																	
RDF2Vec	ΔPr	***-0.15***	0.0	***-0.395***	0.0	***-0.107***	0.0	***-0.292***	0.0	***-0.306***	0.0	***-0.444***	-0.001	-0.031	0.0	***-0.092***	-0.003
	ΔRe	***-0.246***	0.0	***-0.45***	0.0	***-0.091***	0.0	***-0.3***	0.0	***-0.289***	0.0	***-0.505***	-0.004	***-0.01***	0.0	***-0.12***	-0.001
	ΔF1	***-0.253***	0.0	***-0.463***	0.0	***-0.079***	0.0	***-0.298***	0.0	***-0.304***	0.0	***-0.488***	-0.003	-0.008	0.0	***-0.084***	-0.001
	ΔAc	***-0.246***	0.0	***-0.45***	0.0	***-0.091***	0.0	***-0.3***	0.0	***-0.289***	0.0	***-0.505***	-0.004	***-0.01***	0.0	***-0.12***	-0.001
ComplEx	ΔPr	***-0.207***	0.0	***-0.531***	-0.011	-0.046	0.0	***-0.119***	0.0	***-0.224***	0.0	***-0.383***	-0.007	-0.01	0.0	***-0.07***	-0.001
	ΔRe	***-0.286***	0.0	***-0.513***	-0.011	-0.029	0.0	***-0.132***	0.0	***-0.187***	0.0	***-0.368***	-0.005	***-0.023***	0.0	***-0.113***	-0.001
	ΔF1	***-0.301***	0.0	***-0.538***	-0.011	-0.002	0.0	***-0.092***	0.0	***-0.215***	0.0	***-0.359***	-0.005	***-0.022***	0.0	***-0.094***	-0.001
	ΔAc	***-0.286***	0.0	***-0.513***	-0.011	-0.029	0.0	***-0.132***	0.0	***-0.187***	0.0	***-0.368***	-0.005	***-0.023***	0.0	***-0.113***	-0.001
distMult	ΔPr	***-0.162***	0.0	***-0.37***	0.0	-0.003	0.0	***-0.186***	0.0	***-0.277***	0.0	***-0.457***	-0.001	-0.007	0.0	-0.048	-0.01
	ΔRe	***-0.228***	0.0	***-0.438***	0.0	0.003	0.0	***-0.209***	0.0	***-0.204***	0.0	***-0.425***	-0.001	***-0.014***	0.0	***-0.102***	-0.001
	ΔF1	***-0.236***	0.0	***-0.463***	0.0	0.021	0.0	***-0.173***	0.0	***-0.242***	0.0	***-0.431***	-0.001	***-0.012***	0.0	***-0.086***	-0.001
	ΔAc	***-0.228***	0.0	***-0.438***	0.0	0.003	0.0	***-0.209***	0.0	***-0.204***	0.0	***-0.425***	-0.001	***-0.014***	0.0	***-0.102***	-0.001
Sufficient																	
RDF2Vec	ΔPr	***0.0***	-0.933	***0.0***	-0.912	***0.0***	-0.42	***0.0***	-0.42	*-0.023*	-0.718	***-0.001***	-0.708	*-0.038*	-0.396	***-0.001***	-0.387
	ΔRe	***0.0***	-0.92	***0.0***	-0.914	***0.0***	-0.394	***0.0***	-0.394	*-0.026*	-0.7	***-0.003***	-0.703	*-0.002*	-0.538	***-0.008***	-0.537
	ΔF1	***0.0***	-0.919	***0.0***	-0.91	***0.0***	-0.409	***0.0***	-0.409	*-0.029*	-0.704	***-0.003***	-0.703	*-0.006*	-0.451	***-0.001***	-0.45
	ΔAc	***0.0***	-0.92	***0.0***	-0.914	***0.0***	-0.394	***0.0***	-0.394	*-0.026*	-0.7	***-0.003***	-0.703	*-0.002*	-0.538	***-0.001***	-0.537
ComplEx	ΔPr	***0.0***	-0.802	***0.0***	-0.804	***0.0***	-0.422	***0.0***	-0.422	*-0.189*	-0.624	***0.0***	-0.612	*-0.03*	-0.368	***0.0***	-0.368
	ΔRe	***0.0***	-0.812	***0.0***	-0.807	***0.0***	-0.374	***0.0***	-0.374	*-0.106*	-0.618	***0.0***	-0.615	***-0.001***	-0.545	***0.0***	-0.544
	ΔF1	***0.0***	-0.792	***0.0***	-0.79	***0.0***	-0.401	***0.0***	-0.401	*-0.155*	-0.588	***0.0***	-0.586	*-0.002*	-0.456	***0.0***	-0.456
	ΔAc	***0.0***	-0.812	***0.0***	-0.807	***0.0***	-0.374	***0.0***	-0.374	*-0.106*	-0.618	***0.0***	-0.615	***-0.001***	-0.545	***0.0***	-0.544
distMult	ΔPr	***-0.012***	-0.873	***0.0***	-0.849	***0.0***	-0.364	***0.0***	-0.364	*-0.175*	-0.595	***0.0***	-0.597	***-0.021***	-0.398	***0.0***	-0.404
	ΔRe	***-0.011***	-0.863	***0.0***	-0.857	***0.0***	-0.347	***0.0***	-0.347	*-0.106*	-0.549	***0.0***	-0.553	***0.0***	-0.566	***0.0***	-0.567
	ΔF1	***-0.014***	-0.86	***0.0***	-0.85	***0.0***	-0.357	***0.0***	-0.357	*-0.148*	-0.563	***0.0***	-0.567	***-0.001***	-0.481	***0.0***	-0.481
	ΔAc	***-0.011***	-0.863	***0.0***	-0.857	***0.0***	-0.347	***0.0***	-0.347	*-0.106*	-0.549	***0.0***	-0.553	***0.0***	-0.566	***0.0***	-0.567

and MDGENRE.

The length of the explanation and the number of features used to construct the explanation are measures of the quality of explanations [54], related to the idea of sparsity in Explainable AI and the concern that shorter and simpler explanations are preferred and cognitively easier for the human user. No hard evidence was found for preferred sizes for these metrics although a number around 7 is loosely mentioned by Rosenfeld [54] as a reference point based on another work [37]. Figure 5.8 shows the mean and standard deviation for each dataset and type of counterfactual explanation. The mean varies between 2 and 4 facts for the necessary explanations and as expected is larger for the datasets that proved more difficult, the MUTAG and MDGENRE. Notably, sufficient explanations almost always need a single fact to produce satisfactory explanations. These results are promising in terms of generating successful explanations that are also easy to interpret.

Regarding the quality of explanations established by Rosenfeld [54] it is also worth mentioning that no comparison was done between the black-box model being explained and the best performing interpretable model because, according to the conducted literature review there are no reference methods that are interpretable for Node Classification with KGE [1].

[1] Although it could in principle be accomplished by combining an interpretable by design KGE method designed for link

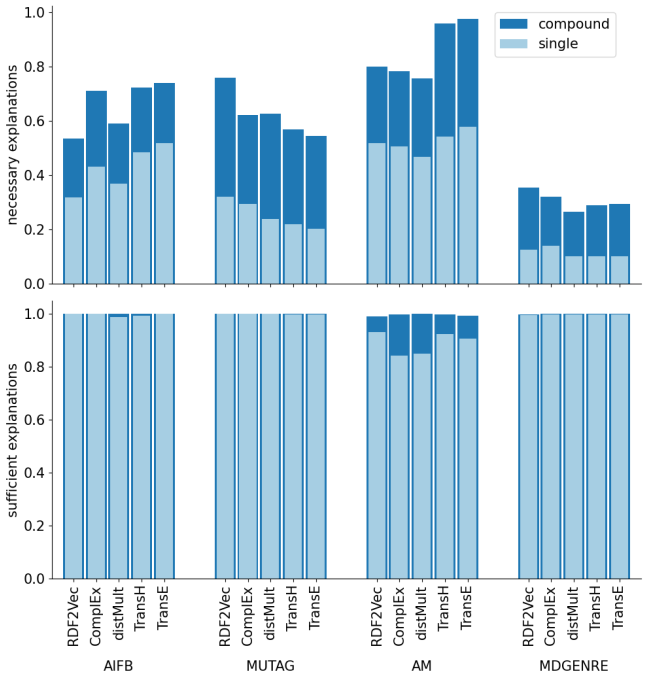

Figure 5.7: Ratio of satisfied necessary and sufficient explanation conditions using the class change condition with C-KEE, for all datasets and KGE models, using RandomForest.

5.4 Example Explanations

To help materialize the application of the explainers, this section gives some concrete examples of explanations given by the C-KEE explainer and shown in Figure 5.9. The examples are retrieved from explanations for predictions on the AIFB and AM datasets. For easier understanding, the original entity names are shortened and, when needed, translated from the original German or Dutch.

Example 1

The chart on the left is generated for an explanation on the AM (Amsterdam Museum) dataset and concerns the classification of an entity labeled *[...] Description of the beautiful temple [...]* as a *book collection*. C-KEE finds two necessary single explanations: the entities labelled *Luyken collection*, with an explanatory power of 0.125, and *book* with an explanatory power of 0.989. These explanations are very sensible when considering that *Luyken collection* refers to the poet and illustrator Jan Luyken, and that a *book collection* is necessarily composed of *books*. The explanatory power is also intuitive since a poet and illustrator may be associated to a book collection but not as much as a book itself.

Example 2

The next example illustrates a node classification belonging to the AIFB dataset where the academic *id2105* is classified as belonging to *Research group 4*. C-KEE finds a necessary compound explanation with an explanatory power of 0.558, showing that a combination of neighbours is needed to explain the prediction. A research project labelled *Projekte id2* is firstly selected but fails to meet the explanation

prediction with an interpretable ML model

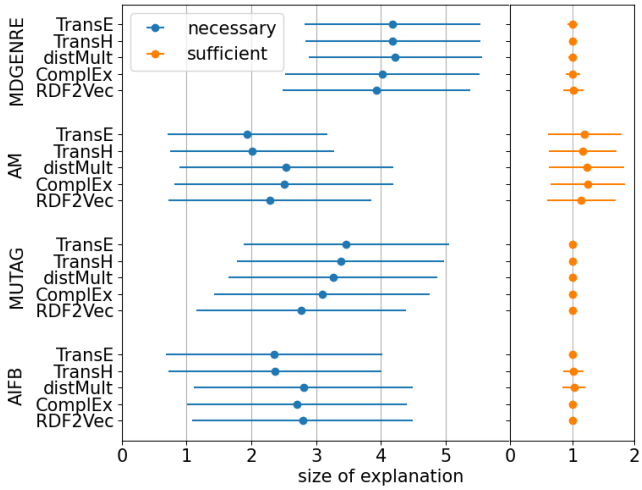

Figure 5.8: Length of explanations (mean and standard deviation) for necessary and sufficient explanations using the class change condition with C-KEE, for all datasets and KGE models, using RandomForest.

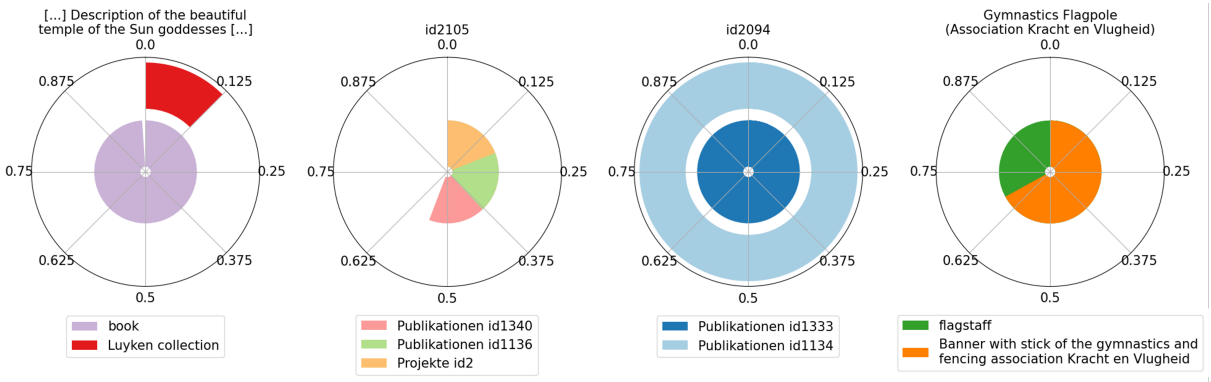

Figure 5.9: Explanations for four node classifications from AIFB and AM. Left to right: two single necessary explanations for the prediction *book collection*, one compound necessary explanation for the prediction *Research group 4*, two sufficient explanations for the prediction *Research group 4*, and one compound sufficient explanation for the prediction *textile collection*.

condition, so two more neighbours corresponding to publications are eventually required to produce a valid explanation. The explanation illustrates that combining several triples may reduce uncertainty – a project may have links to different research groups, so the additional information provided by the publications likely served to disambiguate the prediction.

In fact, due to the simpler nature of the AIFB dataset, it is possible to, a posteriori, explore the graph and find that *Projekte/id2* is connected to many more persons, 2-hops and 3-hops away, that are affiliated with different *research groups* including but not limited to the correct one pointing to the need for additional information.

Example 3

Next, another explanation for a prediction on the AIFB dataset shows that, to correctly classify entity *id2094*, either *Publikationen id1333* or *Publikationen id1134* are sufficient to support a correct prediction by the model.

In a subsequent exploration of the KG it is found that the persons associated with the *Publikationen*

are themselves affiliated with the predicted *Research_group* and thus the given explanation is sufficient for the model to predict the correct class.

Example 4

Finally, a compound sufficient explanation where the combination of AM entities *flagstaff* and *Banner [...] of the gymnastics [...]* is required to correctly classify the entity *Gymnastics Flagpole [...]* as part of a *textile collection* with a combined explanatory power of 1.212.

Chapter 6

Conclusion

6.1 Summary

Explainability solutions have only increased in popularity in recent times. The increased use of AI, the expansion to new and more demanding areas, and the increasingly trivialized use of solutions that are black-box have all contributed to a demand for interpretability. Knowledge Graph Embeddings solutions have been very successful and their application to Node Classification problems in the areas of biomedical and healthcare.

This work addressed a common framework for prediction composed of KGE and traditional SL models for classification that usually lack interpretable capabilities. With a focus on local and model agnostic explanations, two new explainability methods are proposed based on the very successful XAI paradigm of perturbation-based explanations, to directly solve a problem that was yet to be addressed by the relevant literature: LoFI and C-KEE.

Both methods provide counterfactual explanations, allowing future users to correlate small collections of facts present in the Knowledge Graph with the predictions made by the black-box models. Such types of explanations are easier to reason about and thus may improve the experience of the target users for such models. LoFI explores online/update methods to address the challenge of generating explanations faster. However, the update methods allowed an approximation of the behaviour of some entity if perturbed, as opposed to an exact behaviour that could be derived only by re-training from scratch. C-KEE implements an explanation method that completely avoids the need for partial or complete re-training by incorporating an alternative entity representation coupled with a perturbation approach that takes advantage of that representation. Any approach that allows a user to extract valuable explanations in a practical setting with limited time is a valuable addition and this is what was achieved with both methods, but more notably with C-KEE. It was also observed that Node Classification and KGE models used in conjunction with LoFI should allow for more meaningful explanations when compared to GNN and GNN explainers.

In summary, this work showed that it is possible to extract meaningful explanations, and in reasonable time, for a previously untackled problem that is very relevant in important fields of application.

6.2 Contributions

The major contribution of this work is the design, implementation and validation of two novel methods for explainability, the first of a kind able to explain models that use Supervised Learning models and Knowledge Graph Embeddings for Node Classification.

The production of these methods also contributed to:

- Establish the notion that 1-hop facts can be used as a comprehensive representation of an entity in the context of explainability;
- Investigate the application of the concepts of necessary and sufficient conditions to Node Classification with KGEs;
- With LoFI: experiment and validate the use of online/update methods for KGEs, implemented from scratch or in this case already existent (as was the case with Word2vec) as a means to implement perturbations that can be used for explainability purposes in Node Classification with KGEs; and
- With C-KEE: experiment and validate the use of end-to-end explainability frameworks that integrate superior prediction and explainability capabilities.
- One research paper produced, submitted for peer-review.

6.3 Limitations and Future Work

Additional work could be done to further investigate the reasons behind the less successful results using necessary explanations with LoFI. Ablation studies could be performed to improve the understanding of how the different datasets impact the predictive models and explainability methods regarding the quality of the explanations. The results could also be enriched with user studies that could help validate and explore new aspects of the developed explainability methods.

With the extremely successful results obtained using the sufficient explanations and because sufficient explanations are closely related to the idea of a synthesized version of the entity they are explaining, a very interesting possibility is to use those explanations to obtain prototype explanations where it would be possible to summarize what facts or types of facts best describe an entity for some predicted class, thus building on what has been accomplished in order to implement a global scope explanation method.

References

[1] Amina Adadi and Mohammed Berrada. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018. Conference Name: IEEE Access.

[2] Subhan Ali, Filza Akhlaq, Ali Shariq Imran, Zenun Kastrati, Sher Muhammad Daudpota, and Muhammad Moosa. The enlightening role of explainable artificial intelligence in medical & healthcare domains: A systematic literature review. *Computers in Biology and Medicine*, 166:107555, November 2023.

[3] Peter Bloem, Xander Wilcke, Lucas van Berkel, and Victor de Boer. kgbench: A Collection of Knowledge Graph Datasets for Evaluating Relational and Multimodal Machine Learning. In *The Semantic Web: 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings*, pages 614–630, Berlin, Heidelberg, June 2021. Springer-Verlag.

[4] Angana Borah, Manash Pratim Barman, and Amit Awekar. Are Word Embedding Methods Stable and Should We Care About It? *Proceedings of the 32st ACM Conference on Hypertext and Social Media*, pages 45–55, August 2021. Conference Name: HT '21: 32nd ACM Conference on Hypertext and Social Media ISBN: 9781450385510 Place: Virtual Event USA Publisher: ACM.

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[6] Leo Breiman. Random Forests. *Machine Language*, 45(1):5–32, October 2001.

[7] Ruth M. J. Byrne. Counterfactuals in Explainable Artificial Intelligence (XAI): Evidence from Human Reasoning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 6276–6282. International Joint Conferences on Artificial Intelligence Organization, July 2019.

[8] HongYun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1616–1637, September 2018. Conference Name: IEEE Transactions on Knowledge and Data Engineering.

[9] Ricardo M. S. Carvalho, Daniela Oliveira, and Catia Pesquita. Knowledge Graph Embeddings for ICU readmission prediction. *BMC Medical Informatics and Decision Making*, 23(1):12, January 2023.

[10] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, August 2016. Association for Computing Machinery.

[11] David Jaime Tena Cucala, Bernardo Cuenca Grau, Egor V. Kostylev, and Boris Motik. Explainable GNN-Based Models over Knowledge Graphs. In *International Conference on Learning Representations*, March 2022.

[12] S. Decker, S. Melnik, F. van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks. The Semantic Web: the roles of XML and RDF. *IEEE Internet Computing*, 4(5):63–73, September 2000. Conference Name: IEEE Internet Computing.

[13] Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. Explainable artificial intelligence: A survey. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 0210–0215, May 2018.

[14] Alexandre Duval and Fragkiskos D. Malliaros. GraphSVX: Shapley Value Explanations for Graph Neural Networks. In *Machine Learning and Knowledge Discovery in Databases. Research Track: European Conference, ECML PKDD 2021, Bilbao, Spain, September 13–17, 2021, Proceedings, Part II*, pages 302–318, Berlin, Heidelberg, September 2021. Springer-Verlag.

[15] Lisa Ehrlinger and Wolfram Wöß. Towards a Definition of Knowledge Graphs. In *SEMANTICS 2016: Posters and Demos Track*, Leipzig, Germany, 2016. CEUR Workshop Proceedings, vol. 1695. Leipzig: CEUR-WS.org; 2016.

[16] Matthias Fey and Jan Eric Lenssen. Fast Graph Representation Learning with PyTorch Geometric, April 2019. arXiv:1903.02428.

[17] Randy Goebel, Ajay Chander, Katharina Holzinger, Freddy Lecue, Zeynep Akata, Simone Stumpf, Peter Kieseberg, and Andreas Holzinger. Explainable AI: The New 42? In Andreas Holzinger, Peter Kieseberg, A Min Tjoa, and Edgar Weippl, editors, *Machine Learning and Knowledge Extraction*, Lecture Notes in Computer Science, pages 295–303, Cham, 2018. Springer International Publishing.

[18] Aditya Grover and Jure Leskovec. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 855–864, New York, NY, USA, August 2016. Association for Computing Machinery.

[19] Riccardo Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, April 2022.

[20] Nicholas Halliwell, Fabien Gandon, and Freddy Lecue. Linked Data Ground Truth for Quantitative and Qualitative Evaluation of Explanations for Relational Graph Convolutional Network Link Prediction on Knowledge Graphs. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, WI-IAT '21, pages 178–185, New York, NY, USA, April 2022. Association for Computing Machinery.

[21] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 1025–1035, Red Hook, NY, USA, December 2017. Curran Associates Inc.

[22] Yuan He, Jiaoyan Chen, Hang Dong, Ernesto Jiménez-Ruiz, Ali Hadian, and Ian Horrocks. Machine Learning-Friendly Biomedical Datasets for Equivalence and Subsumption Ontology Matching. In Ulrike Sattler, Aidan Hogan, Maria Keet, Valentina Presutti, João Paulo A. Almeida, Hideaki Takeda, Pierre Monnin, Giuseppe Pirrò, and Claudia d'Amato, editors, *The Semantic Web – ISWC 2022*, Lecture Notes in Computer Science, pages 575–591, Cham, 2022. Springer International Publishing.

[23] Farzaneh Heidari and Manos Papagelis. Evolving network representation learning based on random walks. *Applied Network Science*, 5(1):1–38, December 2020. Number: 1 Publisher: SpringerOpen.

[24] Robert R. Hoffman, Shane T. Mueller, Gary Klein, and Jordan Litman. Measures for explainable AI: Explanation goodness, user satisfaction, mental models, curiosity, trust, and human-AI performance. *Frontiers in Computer Science*, 5, 2023.

[25] Aidan Hogan, Claudio Gutierrez, Michael Cochez, Gerard De Melo, Sabrina Kirrane, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Lukas Schmelzeisen, Steffen Staab, Eva Blomqvist, Claudia d'Amato, José Emilio Labra Gayo, Sebastian Neumaier, Anisa Rula, Juan Sequeda, and Antoine Zimmermann. *Knowledge Graphs*. Synthesis Lectures on Data, Semantics, and Knowledge. Springer International Publishing, Cham, 2022.

[26] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: datasets for machine learning on graphs. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, pages 22118–22133, Red Hook, NY, USA, December 2020. Curran Associates Inc.

[27] Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. GraphLIME: Local Interpretable Model Explanations for Graph Neural Networks, September 2020. arXiv:2001.06216.

[28] Maksims Ivanovs, Roberts Kadikis, and Kaspars Ozols. Perturbation-based methods for explaining deep neural networks: A survey. *Pattern Recognition Letters*, 150:228–234, October 2021.

[29] José Jiménez-Luna, Francesca Grisoni, and Gisbert Schneider. Drug discovery with explainable artificial intelligence. *Nature Machine Intelligence*, 2(10):573–584, October 2020. Number: 10 Publisher: Nature Publishing Group.

[30] Bo Kang, Jefrey Lijffijt, and Tijl De Bie. ExplaiNE: An Approach for Explaining Network Embedding-based Link Predictions, April 2019. arXiv:1904.12694.

[31] Hyunju Kang and Hogun Park. Providing Node-level Local Explanation for node2vec through Reinforcement Learning, 2022.

[32] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks, February 2017. arXiv:1609.02907.

[33] Xuan Lin, Zhe Quan, Zhi-Jie Wang, Tengfei Ma, and Xiangxiang Zeng. KGNN: knowledge graph neural network for drug-drug interaction prediction. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20, pages 2739–2745, Yokohama, Yokohama, Japan, January 2021.

[34] Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[35] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943. Publisher: Springer.

[36] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, September 2013. arXiv:1301.3781.

[37] George A. Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2):81–97, 1956. Place: US Publisher: American Psychological Association.

[38] Ramaravind Kommiya Mothilal, Divyat Mahajan, Chenhao Tan, and Amit Sharma. Towards Unifying Feature Attribution and Counterfactual Explanations: Different Means to the Same End. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '21, pages 652–663, New York, NY, USA, July 2021. Association for Computing Machinery.

[39] Susana Nunes, Rita T. Sousa, and Catia Pesquita. Multi-domain knowledge graph embeddings for gene-disease association prediction. *Journal of Biomedical Semantics*, 14(1):11, August 2023.

[40] Osonde A Osoba, William Welser IV, and William Welser. *An intelligence in our image: The risks of bias and errors in artificial intelligence*. Rand Corporation, 2017.

[41] Heiko Paulheim and Johannes Fümkranz. Unsupervised generation of data mining features from linked open data. In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, pages 1–12, New York, NY, USA, June 2012. Association for Computing Machinery.

[42] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.

[43] Phillip E. Pope, Soheil Kolouri, Mohammad Rostami, Charles E. Martin, and Heiko Hoffmann. Explainability Methods for Graph Convolutional Neural Networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10764–10773, June 2019. ISSN: 2575-7075.

[44] Jan Portisch, Nicolas Heist, and Heiko Paulheim. Knowledge graph embedding for data mining vs. knowledge graph embedding for link prediction – two sides of the same coin? *Semantic Web*, 13(3):399–422, 2022. Number: 3 Place: Amsterdam Publisher: IOS Press.

[45] Jan Portisch, Michael Hladik, and Heiko Paulheim. RDF2Vec Light – A Lightweight Approach for Knowledge Graph Embeddings, September 2020. arXiv:2009.07659.

[46] Lara Quijano-Sánchez, Iván Cantador, María E. Cortés-Cediel, and Olga Gil. Recommender systems for smart cities. *Information Systems*, 92:101545, September 2020.

[47] M. Atif Qureshi and Derek Greene. EVE: explainable vector based embedding technique using Wikipedia. *Journal of Intelligent Information Systems*, 53(1):137–165, August 2019.

[48] Arun Rai. Explainable AI: from black box to glass box. *Journal of the Academy of Marketing Science*, 48(1):137–141, January 2020.

[49] Enayat Rajabi and Somayeh Kafaie. Knowledge Graphs and Explainable AI in Healthcare. *Information*, 13(10):459, October 2022. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute.

[50] Mandeep Rathee, Thorben Funke, Avishek Anand, and Megha Khosla. BAGEL: A Benchmark for Assessing Graph Neural Network Explanations, June 2022. arXiv:2206.13983.

[51] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA, August 2016. Association for Computing Machinery.

[52] Petar Ristoski, Gerben Klaas Dirk de Vries, and Heiko Paulheim. A Collection of Benchmark Datasets for Systematic Evaluations of Machine Learning on the Semantic Web. In *The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part II*, pages 186–194, Berlin, Heidelberg, October 2016. Springer-Verlag.

[53] Petar Ristoski and Heiko Paulheim. RDF2Vec: RDF Graph Embeddings for Data Mining. In Paul Groth, Elena Simperl, Alasdair Gray, Marta Sabou, Markus Krötzsch, Freddy Lecue, Fabian Flöck, and Yolanda Gil, editors, *The Semantic Web – ISWC 2016*, Lecture Notes in Computer Science, pages 498–514, Cham, 2016. Springer International Publishing.

[54] Avi Rosenfeld. Better Metrics for Evaluating Explainable Artificial Intelligence. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '21, pages 45–50, Richland, SC, May 2021. International Foundation for Autonomous Agents and Multiagent Systems.

[55] Andrea Rossi, Donatella Firmani, Paolo Merialdo, and Tommaso Teofili. Explaining Link Prediction Systems based on Knowledge Graph Embeddings. In *Proceedings of the 2022 International Conference on Management of Data*, SIGMOD '22, pages 2062–2075, New York, NY, USA, June 2022. Association for Computing Machinery.

[56] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.

[57] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling Relational Data with Graph Convolutional Networks. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web*, Lecture Notes in Computer Science, pages 593–607, Cham, 2018. Springer International Publishing.

[58] Thomas Schnake, Oliver Eberle, Jonas Lederer, Shinichi Nakajima, Kristof T. Schütt, Klaus-Robert Müller, and Grégoire Montavon. Higher-Order Explanations of Graph Neural Networks via Relevant Walks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7581–7596, November 2022. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[59] Simon Schramm, Christoph Wehner, and Ute Schmid. Comprehensible Artificial Intelligence on Knowledge Graphs: A survey. *Journal of Web Semantics*, 79:100806, December 2023.

[60] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, April 2014. arXiv:1312.6034.

[61] Rita T. Sousa, Sara Silva, and Catia Pesquita. Evolving knowledge graph similarity for supervised learning in complex biomedical domains. *BMC Bioinformatics*, 21(1):6, January 2020.

[62] Rita T. Sousa, Sara Silva, and Catia Pesquita. Supervised Semantic Similarity, May 2021. Pages: 2021.02.16.431402 Section: New Results.

[63] Rita T. Sousa, Sara Silva, and Catia Pesquita. Explainable Representations for Relation Prediction in Knowledge Graphs. *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, 19(1):635–646, August 2023. Conference Name: Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning.

[64] Bram Steenwinckel, Gilles Vandewiele, Michael Weyns, Terencio Agozzino, Filip De Turck, and Femke Ongenae. INK: knowledge graph embeddings for node classification. *Data Mining and Knowledge Discovery*, 36(2):620–667, March 2022.

[65] Michael Cheng-Tek Tai. The impact of artificial intelligence on human society and bioethics. *Tzu-Chi Medical Journal*, 32(4):339–343, August 2020.

[66] Ilaria Tiddi, Freddy Lécué, and Pascal Hitzler, editors. *Knowledge Graphs for eXplainable Artificial Intelligence: Foundations, Applications and Challenges*, volume 47 of *Studies on the Semantic Web*. IOS Press, 2020.

[67] Ilaria Tiddi and Stefan Schlobach. Knowledge graphs as tools for explainable machine learning: A survey. *Artificial Intelligence*, 302:103627, January 2022.

[68] Richard Tomsett, Dave Braines, Dan Harborne, Alun Preece, and Supriyo Chakraborty. Interpretable to Whom? A Role-based Model for Analyzing Interpretable Machine Learning Systems, June 2018. arXiv:1806.07552.

[69] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. Complex Embeddings for Simple Link Prediction. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 2071–2080. PMLR, June 2016. ISSN: 1938-7228.

[70] Warren J. von Eschenbach. Transparency and the Black Box Problem: Why We Do Not Trust AI. *Philosophy & Technology*, 34(4):1607–1622, December 2021.

[71] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, September 2014.

[72] Joel Walmsley. Artificial intelligence and the value of transparency. *AI & SOCIETY*, 36(2):585–595, June 2021.

[73] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural Deep Network Embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 1225–1234, San Francisco, California, USA, 2016. ACM Press.

[74] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, December 2017.

[75] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge Graph Embedding by Translating on Hyperplanes. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1), June 2014. Number: 1.

[76] David S. Watson, Limor Gultchin, Ankur Taly, and Luciano Floridi. Local Explanations via Necessity and Sufficiency: Unifying Theory and Practice. *Minds and Machines*, 32(1):185–218, March 2022.

[77] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding Entities and Relations for Learning and Inference in Knowledge Bases, August 2015. arXiv:1412.6575.

[78] Zi Ye, Yogan Jaya Kumar, Goh Ong Sing, Fengyan Song, and Junsong Wang. A Comprehensive Survey of Graph Neural Networks for Knowledge Graphs. *IEEE Access*, 10:75729–75741, 2022. Conference Name: IEEE Access.

[79] Chih-Kuan Yeh, Been Kim, and Pradeep Ravikumar. Chapter 15. Human-Centered Concept Explanations for Neural Networks. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, pages 337–352. IOS Press, 2021.

[80] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. GNNExplainer: Generating Explanations for Graph Neural Networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[81] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. XGNN: Towards Model-Level Explanations of Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pages 430–438, New York, NY, USA, August 2020. Association for Computing Machinery.

[82] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in Graph Neural Networks: A Taxonomic Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5782–5799, May 2023. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[83] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On Explainability of Graph Neural Networks via Subgraph Explorations. In *Proceedings of the 38th International Conference on Machine Learning*, pages 12241–12252. PMLR, July 2021. ISSN: 2640-3498.

[84] Da Zheng, Minjie Wang, Quan Gan, Xiang Song, Zheng Zhang, and George Karypis. Scalable Graph Neural Networks with Deep Graph Library. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, pages 1141–1142, New York, NY, USA, March 2021. Association for Computing Machinery.

Appendix A

Training Results for the KGE Model Parameters

Table A.1: Top-5 Grid Search Cross Validation results for RDF2Vec parameters and base classifier for the AIFB dataset.

vector_size	sg	max_depth	max_walks	base classifier	RDF2Vec get_walks (s) (mean)	RDF2Vec fit (s) (mean)	NC fit (s) (mean)	Accuracy (mean)	F1-Score (mean)
500	1	4	500	RandomForest-Classifier	2.331495	4.749773	0.362728	0.907143	0.902891
500	0	4	500	SVC	2.312124	1.903030	0.040938	0.900000	0.894540
50	0	4	500	SVC	2.312570	1.115656	0.022047	0.900000	0.894540
200	0	4	500	SVC	2.300040	1.252389	0.031419	0.900000	0.894540
50	1	4	500	RandomForest-Classifier	0.967186	1.080194	0.274197	0.900000	0.894017

Table A.2: Top-5 Grid Search Cross Validation results for RDF2Vec parameters and base classifier for the MUTAG dataset.

vector_size	sg	max_depth	max_walks	base classifier	RDF2Vec get_walks (s) (mean)	RDF2Vec fit (s) (mean)	NC fit (s) (mean)	Accuracy (mean)	F1-Score (mean)
50	1	4	500	RandomForest-Classifier	2.448547	2.280855	0.379987	0.746263	0.737473
50	1	2	500	GaussianNB	1.604026	1.171228	0.007500	0.735286	0.730375
50	1	4	500	SVC	2.448547	2.280855	0.027609	0.742896	0.724862
500	1	4	500	RandomForest-Classifier	2.503333	5.289215	0.696783	0.731650	0.719818
50	1	2	500	RandomForest-Classifier	1.604026	1.171228	0.371073	0.731650	0.717143

Table A.3: Top-5 Grid Search Cross Validation results for RDF2Vec parameters and base classifier for the AM dataset.

vector_size	sg	max_depth	max_walks	base classifier	RDF2Vec get_walks (s) (mean)	RDF2Vec fit (s) (mean)	NC fit (s) (mean)	Accuracy (mean)	F1-Score (mean)
500	1	2	500	RandomForest-Classifier	40.412594	13.141304	1.793141	0.774363	0.763723
200	1	2	500	RandomForest-Classifier	40.427874	8.395103	1.274717	0.766879	0.755961
50	1	2	500	SVC	40.376448	5.587741	0.263446	0.766856	0.755363
200	1	2	500	SVC	40.427874	8.395103	0.449046	0.770629	0.753645
50	1	2	500	RandomForest-Classifier	40.376448	5.587741	0.813242	0.761825	0.749096

Table A.4: Top-5 Grid Search Cross Validation results for RDF2Vec parameters and base classifier for the MDGENRE dataset.

vector_size	sg	max_depth	max_walks	base classifier	RDF2Vec get_walks (s) (mean)	RDF2Vec fit (s) (mean)	NC fit (s) (mean)	Accuracy (mean)	F1-Score (mean)
200	1	2	500	SVC	35.735697	107.921864	13.881442	0.663287	0.566052
500	1	2	500	SVC	35.779949	166.885115	22.101799	0.662768	0.565061
50	1	2	500	SVC	35.385395	75.332860	5.957358	0.659908	0.561888
200	1	2	500	SVC	19.481651	56.744562	14.265437	0.651070	0.549617
500	1	2	500	RandomForest-Classifier	35.779949	166.885115	16.702771	0.650024	0.548683

Appendix B

Accuracy Results for the Explainers

B.1 Explainers Evaluation

B.1.1 LoFI Explainer Evaluation

Test Results on Accuracy Scores of Sufficient Explanations

Table B.1: Wilcoxon signed rank statistical test results for the test set for all datasets, for accuracy scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	Sig.	Decision
AIFB	0.008	**Reject the null hypothesis**
MUTAG	0.219	Retain the null hypothesis
AM	0.037	**Reject the null hypothesis**
MDGENRE	0.005	**Reject the null hypothesis**

The same tests employed for F1-scores were carried out for the accuracy scores. Table B.1 shows the decisions and Figure B.1 shows the corresponding plot. The decisions and plots are quite similar to the ones obtained for the F1-scores.

Test Results on Accuracy Scores of Necessary Explanations

Table B.2: Wilcoxon signed rank statistical test results for the test set for all datasets, for accuracy scores, for necessary explanations, for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	Sig.	Decision
AIFB	0.011	Reject the null hypothesis
MUTAG	0.368	Retain the null hypothesis
AM	0.235	Retain the null hypothesis
MDGENRE	0.942	Retain the null hypothesis

The same tests employed for F1-scores were carried out for the accuracy score. Table B.2 shows the decisions obtained from those tests and Figure B.2 shows the plot for the results. The results are quite similar to ones on the main text.

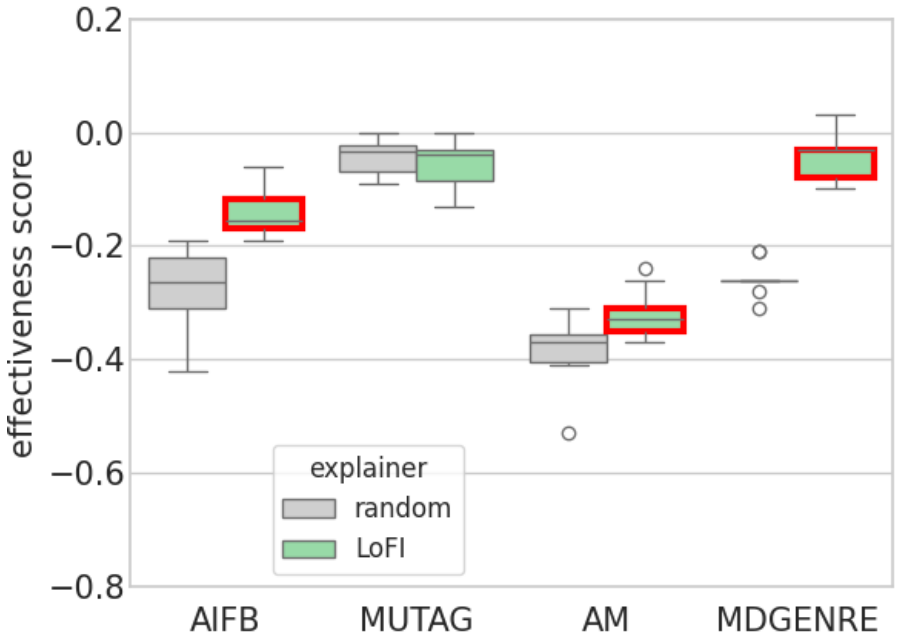

Figure B.1: Results for the test set for all datasets, for accuracy scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

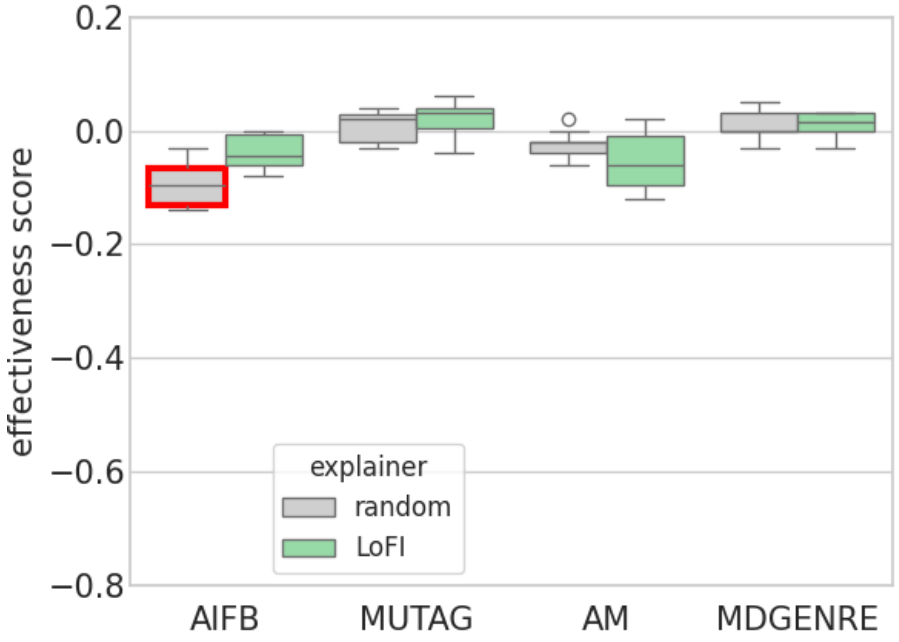

Figure B.2: Results for the test set for all datasets, for accuracy scores, for necessary explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

B.1.2 C-KEE Explainer Evaluation

Test Results on Accuracy Scores of Sufficient Explanations

Table B.3: Wilcoxon signed rank statistical test results for the test set for all datasets, for accuracy scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	Sig.	Decision
AIFB	0.005	**Reject the null hypothesis**
MUTAG	0.046	**Reject the null hypothesis**
AM	0.005	**Reject the null hypothesis**
MDGENRE	0.005	**Reject the null hypothesis**

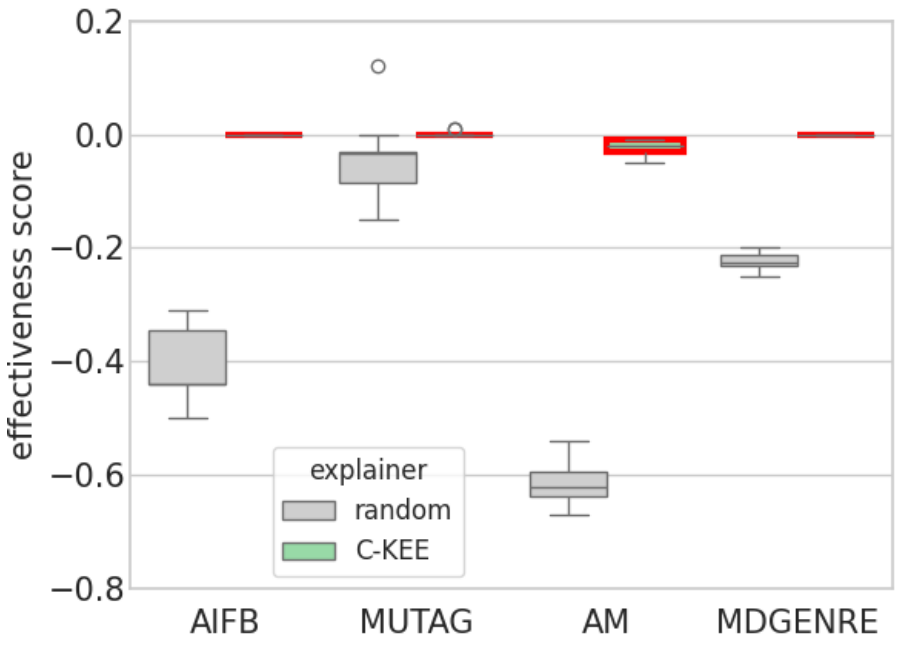

Figure B.3: Results for the test set for all datasets, for accuracy scores, for sufficient explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

Table B.3 shows the decisions and Figure B.3 shows the corresponding plot. The decisions and plots are quite similar to the ones obtained for the F1-scores except that in accuracy results a desirable significant score is also found for the MUTAG dataset.

Test Results on Accuracy Scores of Necessary Explanations

Table B.4 shows the decisions obtained for the statistical tests and Figure B.4 shows the plot for the results. The results are quite similar to the ones for the F1-scores.

Table B.4: Wilcoxon signed rank statistical test results for the test set for all datasets, for accuracy scores, for necessary explanations for maximum explanation length of 5, with 10 independent runs for each model.

Dataset	Sig.	Decision
AIFB	0.005	**Reject the null hypothesis**
MUTAG	0.608	Retain the null hypothesis
AM	0.005	**Reject the null hypothesis**
MDGENRE	0.005	**Reject the null hypothesis**

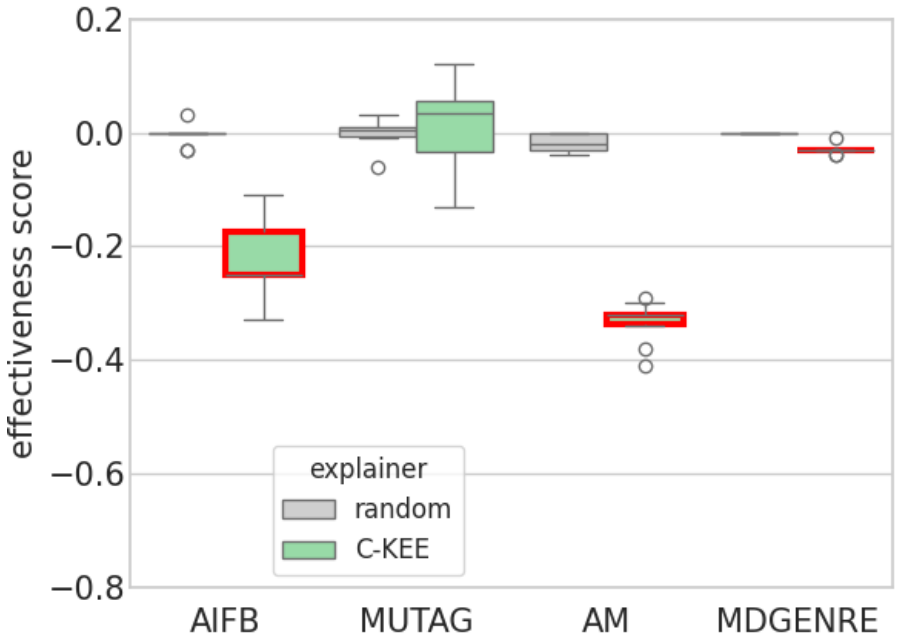

Figure B.4: Results for the test set for all datasets, for accuracy scores, for necessary explanations, for maximum explanation length of 5, with 10 independent runs for each model. Statistically significant results, per dataset, outlined in red.

Table B.5: Mann-Whitney U statistical test results for the test set for all datasets, for accuracy scores, for necessary explanations, for maximum explanation length of 5 - Node Classification with KGE explained with C-KEE and LoFI. Using 10 independent runs for each model.

	AIFB	MUTAG	AM	MDGENRE
C-KEE-LoFI	**<0.001**	**<0.001**	**<0.001**	**0.023**

B.1.3 Comparison of LoFI, C-KE and GNN Solutions - Accuracy Results

Comparison of Sufficient Explanations

Table B.5 shows the decisions obtained from the statistical tests. No relevant differences were found compared to the F1-scores.

Comparison of Necessary Explanations

Table B.6: Mann-Whitney U statistical test results for the test set for all datasets, for accuracy scores, for necessary explanations, for maximum explanation length of 5 - Node Classification with KGE explained with C-KEE and LoFI, Node Classification with GNN explained with GNN explainer models. Using 10 independent runs for each model.

	AIFB	**MUTAG**	**AM**	**MDGENRE**
LoFI-GradExplainer	**0.007**	0.315	**0.004**	0.075
LoFI-GNNExplainer	**0.009**	0.143	0.529	0.029
C-KEE-GradExplainer	**<0.001**	0.529	**<0.001**	**<0.001**
C-KEE-GNNExplainer	**<0.001**	0.280	**<0.001**	0.190
C-KEE-LoFI	**<0.001**	1.000	**<0.001**	**<0.001**

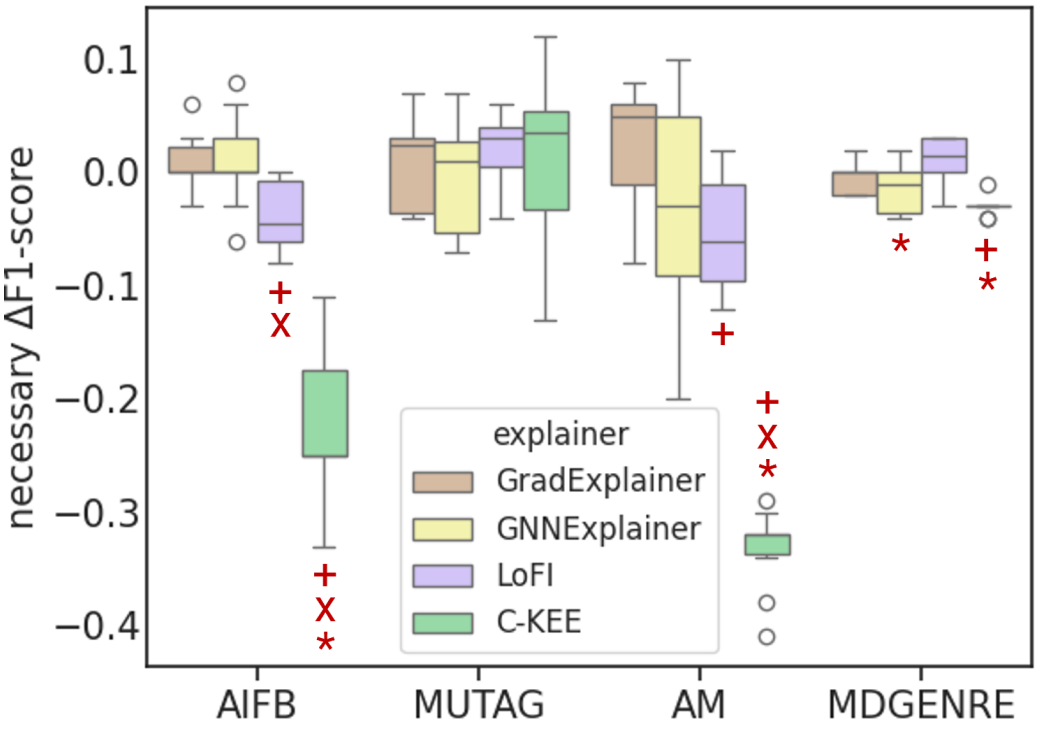

Figure B.5: Results for the test set for all datasets, for accuracy scores, for necessary explanations, for maximum explanation length of 5 - Node Classification with KGE explained with C-KEE and LoFI, Node Classification with GNN explained with GNN explainer models, with 10 independent runs for each model. Statistically significant results, per dataset, highlighted by the red marks: "**+**" better than GradExplainer, "**X**" better than GNNExplainer, "*****" better than LoFI.

Table B.6 shows the decisions obtained from the statistical tests and Figure B.5 shows the plot for

the results. The results are quite similar to the ones for the F1-scores, except that C-KEE explainer outperforms also the GradExplainer solution in MDGENRE.

Appendix C

C-KEE Sufficient Explanation Algorithm

Algorithm 2 details the procedure taken by C-KEE to generate sufficient explanations based on the class change condition, which can be trivially altered to use the class probability condition.

Algorithm 2 Generate sufficient explanations.

Input: entity e;
KGE model K;
node classification model M;
explanation maximum length l;
Output: the set X of explanations

$X \leftarrow \emptyset$
$relevance_{best} \leftarrow 0$
$N \leftarrow$ ALL NEIGHBOURS(e)
$embs \leftarrow$ GET NEIGHBOUR EMBEDDINGS(K, N)
$repr \leftarrow$ AGGREGATE REPRESENTATION$(embs)$
$class_o, class\_prob_o \leftarrow$ PREDICT$(M, repr)$
$curr\_len \leftarrow 0$
$explanation \leftarrow []$
$best\_explanation \leftarrow []$
while $X == \emptyset$ and $curr\_len <= |N|$ and $curr\_len <= l$ **do**
  for $neighb \in N \backslash best\_explanation$ **do**
    $embs' \leftarrow$ GET NEIGHBOUR EMBEDDINGS$(K, best_explanation \cup neighb)$
    $repr' \leftarrow$ AGGREGATE REPRESENTATION$(embs')$
    $class_x, class\_prob_x \leftarrow$ PREDICT$(M, repr')$
    $relevance \leftarrow$ COMPARE$(class_prob_o, class_prob_x)$
    $explanation[curr\_len] = neighb$
    if $class_o = class_x$ **then**
      $X.append(explanation)$
    else if $relevance >= relevance_{best}$ **then**
      $past\_explanation = explanation$
      $relevance_{best} = relevance$
  $best\_explanation = past\_explanation$
  $curr\_len += 1$
if $X = \emptyset$ **then return** $best\_explanation$
else return X

Appendix D

Parameters for the Additional Results with C-KEE

D.1 KG Embedding Methods

For the experiments, five representative KGE methods were implemented: RDF2Vec [53], ComplEx [69], distMult [77], TransH [75] and TransE [5].

Table D.1: RDF2Vec parameters.

Parameter	**Value**
Embedding size	100
Walk depth	2
Maximum number of walks	None
Word2vec model	CBOW

Table D.2: ComplEx, distMult, TransE, TransH parameters.

Parameter	**ComplEx**	**distMul**	**TransE**	**TransH**
Embedding size	100	100	100	100
Optimization	Adagrad	Adagrad	SGD	Adagrad
Train times	1000	500	500	500
Number batches	100	100	100	100
Entity neg rate	1	1	1	1
Relation neg rate	0	0	0	0
Bern	1	1	0	0
Alpha	0.5	0.5	0.001	0.001
Lambda	0.05	0.05	–	–
Margin	–	–	1	1

The experiments use an existing RDF2Vec Python implementation[1] and the OpenKE library[2]. The

[1] https://github.com/IBCNServices/pyRDF2Vec
[2] https://github.com/thunlp/OpenKE/tree/OpenKE-Tensorflow1.0

parameters used for each KGE model are described in Tables D.1 and D.2.

D.2 Supervised Learning models

Table D.3: XGBoost and RandomForest parameters that have been optimized.

Parameter	Values
Maximum depth	2,4,6,8,10

Table D.4: MLP parameters that have been optimized.

Parameter	Values
Hidden layer sizes	(100,), (50,50), (50,100,50)
Activation function	hyperbolic tan function, relu function
Optimization method	SGD, Adam
Alpha	0.0001, 0.05
Learning rate	constant, adaptive

The experiments use two ensemble methods, RandomForest [6] and XGBoost [10], and NN methods presenting different hidden layer configurations, with MLP [56] being the one that achieved the best results. To run these supervised learning methods, scikit-learn [42] is employed to optimize certain parameters. The parameters are supplied in Tables D.3 and D.4.

Appendix E

Additional Results with C-KEE

E.1 Predictive Models Results

Table E.1: Mean and standard deviation of precision, recall, weighted average F1-score and accuracy (Pr, Re, F1, Ac) comparing C-KEE global approach to the baseline when coupled with different KGE methods and the Random Forest classifier. Highlighted results (bold) are statistically significantly better than the direct comparison.

		AIFB		**MUTAG**		**AM**		**MDGENRE**	
		baseline	C-KEE	baseline	C-KEE	baseline	C-KEE	baseline	C-KEE
TransH	Pr	0.569 (0.08)	**0.802 (0.183)**	0.372 (0.089)	**0.576 (0.178)**	0.137 (0.028)	**0.53 (0.118)**	0.421 (0.029)	0.444 (0.029)
	Re	0.658 (0.089)	**0.794 (0.141)**	0.568 (0.024)	0.597 (0.054)	0.357 (0.005)	**0.583 (0.058)**	0.551 (0.001)	**0.62 (0.003)**
	F1	0.581 (0.092)	**0.767 (0.151)**	0.439 (0.046)	**0.518 (0.08)**	0.191 (0.007)	**0.514 (0.07)**	0.395 (0.003)	**0.505 (0.004)**
	Ac	0.658 (0.107)	**0.794 (0.103)**	0.568 (0.071)	0.597 (0.068)	0.357 (0.04)	**0.583 (0.052)**	0.551 (0.019)	**0.62 (0.018)**
TransE	Pr	0.635 (0.148)	**0.828 (0.194)**	0.536 (0.225)	**0.669 (0.18)**	0.129 (0.005)	**0.599 (0.105)**	0.349 (0.034)	**0.439 (0.029)**
	Re	0.704 (0.086)	**0.84 (0.168)**	0.597 (0.024)	**0.644 (0.072)**	0.355 (0.004)	**0.591 (0.091)**	0.548 (0.0)	**0.616 (0.004)**
	F1	0.641 (0.097)	**0.813 (0.178)**	0.473 (0.041)	**0.586 (0.119)**	0.188 (0.005)	**0.539 (0.093)**	0.389 (0.001)	**0.501 (0.004)**
	Ac	0.704 (0.084)	**0.84 (0.102)**	0.597 (0.087)	**0.644 (0.102)**	0.355 (0.037)	**0.591 (0.059)**	0.548 (0.019)	**0.616 (0.021)**

Table E.2: Mean and standard deviation of precision, recall, weighted average F1-score and accuracy (Pr, Re, F1, Ac) comparing C-KEE global approach to the baseline when coupled with different KGE methods and the XGBoost classifier. Highlighted results (bold) are statistically significantly better than the direct comparison.

		AIFB		**MUTAG**		**AM**		**MDGENRE**	
		baseline	C-KEE	baseline	C-KEE	baseline	C-KEE	baseline	C-KEE
RDF2Vec	Pr	0.932 (0.104)	0.944 (0.04)	0.733 (0.136)	0.715 (0.117)	0.736 (0.084)	**0.836 (0.055)**	0.565 (0.06)	0.566 (0.068)
	Re	0.932 (0.093)	0.932 (0.072)	0.715 (0.126)	0.709 (0.11)	0.725 (0.05)	**0.814 (0.073)**	**0.652 (0.022)**	0.639 (0.016)
	F1	0.925 (0.097)	0.929 (0.059)	0.709 (0.134)	0.695 (0.126)	0.712 (0.062)	**0.81 (0.065)**	**0.578 (0.026)**	0.56 (0.021)
	Ac	0.932 (0.059)	0.932 (0.052)	0.715 (0.118)	0.709 (0.105)	0.725 (0.031)	**0.814 (0.039)**	**0.652 (0.017)**	0.639 (0.018)
ComplEx	Pr	0.878 (0.12)	0.859 (0.172)	0.723 (0.095)	0.735 (0.112)	0.538 (0.069)	**0.736 (0.056)**	0.534 (0.042)	0.558 (0.056)
	Re	0.853 (0.089)	0.84 (0.179)	0.709 (0.1)	0.721 (0.116)	0.548 (0.058)	**0.735 (0.055)**	0.631 (0.021)	**0.644 (0.019)**
	F1	0.849 (0.101)	0.834 (0.175)	0.702 (0.105)	0.71 (0.122)	0.521 (0.059)	**0.721 (0.051)**	0.544 (0.021)	**0.556 (0.027)**
	Ac	0.853 (0.047)	0.84 (0.115)	0.709 (0.083)	0.721 (0.104)	0.548 (0.041)	**0.735 (0.029)**	0.631 (0.019)	**0.644 (0.02)**
distMult	Pr	0.913 (0.087)	0.891 (0.128)	0.731 (0.085)	0.723 (0.106)	0.602 (0.063)	**0.798 (0.106)**	0.564 (0.037)	0.57 (0.052)
	Re	0.898 (0.1)	0.874 (0.152)	0.718 (0.071)	0.7 (0.096)	0.601 (0.059)	**0.779 (0.074)**	0.651 (0.02)	**0.658 (0.026)**
	F1	0.896 (0.097)	0.868 (0.144)	0.71 (0.071)	0.692 (0.097)	0.58 (0.053)	**0.771 (0.074)**	0.565 (0.02)	**0.579 (0.031)**
	Ac	0.898 (0.063)	0.874 (0.09)	0.718 (0.065)	0.7 (0.093)	0.601 (0.045)	**0.779 (0.033)**	0.651 (0.021)	**0.658 (0.021)**
TransH	Pr	0.54 (0.106)	**0.876 (0.13)**	0.544 (0.07)	**0.63 (0.115)**	0.251 (0.059)	**0.666 (0.06)**	0.425 (0.027)	0.509 (0.077)
	Re	0.579 (0.098)	**0.857 (0.146)**	0.55 (0.06)	**0.629 (0.105)**	0.354 (0.026)	**0.645 (0.065)**	0.581 (0.003)	**0.627 (0.02)**
	F1	0.54 (0.09)	**0.857 (0.139)**	0.529 (0.078)	0.612 (0.11)	0.268 (0.032)	**0.627 (0.055)**	0.468 (0.004)	**0.532 (0.027)**
	Ac	0.579 (0.109)	**0.857 (0.098)**	0.55 (0.077)	**0.629 (0.103)**	0.354 (0.053)	**0.645 (0.033)**	0.581 (0.023)	**0.627 (0.023)**
TransE	Pr	0.707 (0.162)	**0.843 (0.194)**	0.551 (0.099)	**0.647 (0.094)**	0.233 (0.048)	**0.639 (0.124)**	0.412 (0.018)	**0.518 (0.076)**
	Re	0.705 (0.107)	**0.857 (0.184)**	0.559 (0.089)	**0.638 (0.095)**	0.333 (0.017)	**0.642 (0.086)**	0.573 (0.002)	**0.627 (0.015)**
	F1	0.69 (0.115)	**0.842 (0.188)**	0.537 (0.1)	**0.631 (0.101)**	0.249 (0.027)	**0.621 (0.099)**	0.46 (0.003)	**0.531 (0.022)**
	Ac	0.705 (0.064)	**0.857 (0.106)**	0.559 (0.106)	**0.638 (0.104)**	0.333 (0.039)	**0.642 (0.067)**	0.573 (0.017)	**0.627 (0.019)**

Table E.3: Mean and standard deviation of precision, recall, weighted average F1-score and accuracy (Pr, Re, F1, Ac) comparing C-KEE global approach to the baseline when coupled with different KGE methods and the MLP classifier. Highlighted results (bold) are statistically significantly better than the direct comparison.

		AIFB		**MUTAG**		**AM**		**MDGENRE**	
		baseline	C-KEE	baseline	C-KEE	baseline	C-KEE	baseline	C-KEE
RDF2Vec	Pr	**0.768 (0.087)**	0.629 (0.178)	0.765 (0.118)	0.728 (0.135)	0.779 (0.07)	**0.858 (0.067)**	0.564 (0.043)	0.552 (0.055)
	Re	0.932 (0.074)	0.932 (0.069)	**0.753 (0.097)**	0.638 (0.109)	0.679 (0.046)	**0.765 (0.063)**	0.611 (0.031)	**0.639 (0.032)**
	F1	0.929 (0.06)	0.928 (0.057)	**0.741 (0.107)**	0.61 (0.148)	0.761 (0.067)	**0.844 (0.062)**	0.582 (0.028)	0.573 (0.029)
	Ac	0.932 (0.058)	0.932 (0.052)	**0.753 (0.087)**	0.638 (0.114)	0.77 (0.046)	**0.845 (0.028)**	0.611 (0.028)	**0.639 (0.017)**
ComplEx	Pr	0.82 (0.124)	**0.884 (0.112)**	0.55 (0.102)	**0.658 (0.052)**	0.49 (0.072)	**0.637 (0.069)**	0.528 (0.045)	**0.554 (0.06)**
	Re	0.807 (0.124)	0.83 (0.117)	0.541 (0.091)	**0.647 (0.056)**	0.483 (0.072)	**0.631 (0.05)**	0.615 (0.036)	**0.635 (0.027)**
	F1	0.799 (0.115)	0.833 (0.111)	0.532 (0.089)	**0.641 (0.059)**	0.473 (0.062)	**0.619 (0.049)**	0.557 (0.033)	**0.576 (0.028)**
	Ac	0.807 (0.085)	0.83 (0.079)	0.541 (0.09)	**0.647 (0.057)**	0.483 (0.042)	**0.631 (0.026)**	0.615 (0.019)	**0.635 (0.017)**
distMult	Pr	0.941 (0.056)	0.942 (0.091)	0.677 (0.07)	0.677 (0.093)	0.564 (0.057)	**0.684 (0.076)**	0.542 (0.028)	**0.568 (0.044)**
	Re	0.931 (0.095)	0.92 (0.115)	0.665 (0.087)	0.665 (0.106)	0.56 (0.047)	**0.674 (0.074)**	0.596 (0.02)	**0.633 (0.028)**
	F1	0.929 (0.086)	0.921 (0.109)	0.656 (0.087)	0.661 (0.096)	0.549 (0.048)	**0.667 (0.071)**	0.564 (0.019)	**0.588 (0.029)**
	Ac	0.931 (0.067)	0.92 (0.068)	0.665 (0.079)	0.665 (0.092)	0.56 (0.049)	**0.674 (0.05)**	0.596 (0.015)	**0.633 (0.025)**
TransH	Pr	0.527 (0.191)	**0.861 (0.213)**	0.58 (0.097)	0.571 (0.121)	0.173 (0.037)	**0.353 (0.039)**	0.426 (0.043)	**0.462 (0.041)**
	Re	0.59 (0.138)	**0.874 (0.173)**	0.571 (0.095)	0.582 (0.048)	0.348 (0.017)	**0.465 (0.026)**	0.585 (0.004)	**0.633 (0.013)**
	F1	0.54 (0.149)	**0.858 (0.184)**	0.567 (0.09)	0.556 (0.083)	0.217 (0.018)	**0.389 (0.022)**	0.477 (0.004)	**0.524 (0.018)**
	Ac	0.59 (0.125)	**0.874 (0.097)**	0.571 (0.072)	0.582 (0.066)	0.348 (0.038)	**0.465 (0.042)**	0.585 (0.02)	**0.633 (0.02)**
TransE	Pr	0.634 (0.114)	**0.881 (0.158)**	0.628 (0.091)	0.608 (0.149)	0.146 (0.022)	**0.32 (0.021)**	0.401 (0.018)	**0.468 (0.017)**
	Re	0.704 (0.086)	**0.84 (0.168)**	0.618 (0.082)	0.624 (0.09)	0.341 (0.014)	**0.444 (0.015)**	0.569 (0.007)	**0.634 (0.019)**
	F1	0.642 (0.096)	**0.866 (0.155)**	0.615 (0.086)	0.601 (0.122)	0.199 (0.017)	**0.363 (0.014)**	0.461 (0.008)	**0.528 (0.017)**
	Ac	0.67 (0.111)	**0.881 (0.087)**	0.618 (0.098)	0.624 (0.087)	0.341 (0.041)	**0.444 (0.029)**	0.569 (0.018)	**0.634 (0.023)**

E.2 C-KEE Explainer Model Results

Results for the Remaining KGE Methods and RandomForest Omitted from the Main Text

Table E.4: Mean of explanation effectiveness of necessary and sufficient explanations with class change condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using RandomForest.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$
Necessary																	
TransH	ΔPr	-0.139	0.0	***-0.339***	-0.017	0.045	0.0	-0.026	0.0	-0.066	0.0	***-0.335***	0.01	-0.002	0.0	***-0.079***	-0.0
	ΔRe	***-0.234***	0.0	***-0.393***	-0.006	0.024	0.0	***-0.074***	0.0	***-0.113***	0.0	***-0.438***	0.001	-0.01	0.0	***-0.101***	0.0
	ΔF1	***-0.224***	0.0	***-0.408***	-0.01	0.057	0.0	0.008	0.0	***-0.079***	0.0	***-0.374***	0.003	-0.008	0.0	***-0.084***	0.0
	ΔAc	***-0.234***	0.0	***-0.393***	-0.006	0.024	0.0	***-0.074***	0.0	***-0.113***	0.0	***-0.438***	0.001	-0.01	0.0	***-0.101***	0.0
TransE	ΔPr	***-0.195***	0.0	***-0.533***	0.0	-0.032	0.0	-0.082	0.0	***-0.175***	0.0	***-0.45***	-0.001	0.004	0.0	***-0.069***	-0.0
	ΔRe	***-0.318***	0.0	***-0.534***	0.0	-0.015	0.0	-0.085	0.0	***-0.154***	0.0	***-0.461***	0.0	***-0.008***	0.0	***-0.1***	-0.0
	ΔF1	***-0.327***	0.0	***-0.575***	0.0	0.011	0.0	-0.023	0.0	***-0.138***	0.0	***-0.417***	0.0	***-0.007***	0.0	***-0.083***	-0.0
	ΔAc	***-0.318***	0.0	***-0.534***	0.0	-0.015	0.0	-0.085	0.0	***-0.154***	0.0	***-0.461***	0.0	***-0.008***	0.0	***-0.1***	-0.0
Sufficient																	
TransH	ΔPr	***-0.02***	-0.642	***0.0***	-0.64	***0.004***	-0.149	***0.0***	-0.149	*-0.064*	-0.482	***0.001***	-0.487	***-0.01***	-0.353	***-0.01***	-0.352
	ΔRe	***-0.006***	-0.664	***0.0***	-0.664	***0.003***	-0.194	***0.0***	-0.194	*-0.039*	-0.529	***0.001***	-0.524	***-0.0***	-0.545	***-0.0***	-0.543
	ΔF1	***-0.01***	-0.645	***0.0***	-0.639	***0.002***	-0.211	***0.0***	-0.211	*-0.052*	-0.471	***0.001***	-0.47	***-0.001***	-0.457	***-0.0***	-0.456
	ΔAc	***-0.006***	-0.664	***0.0***	-0.664	***0.003***	-0.194	***0.0***	-0.194	*-0.039*	-0.529	***0.001***	-0.524	***-0.0***	-0.545	***-0.0***	-0.543
TransE	ΔPr	***0.0***	-0.717	***0.0***	-0.716	***0.006***	-0.341	***0.0***	-0.341	*-0.111*	-0.535	***-0.009***	-0.535	***-0.008***	-0.321	***-0.008***	-0.325
	ΔRe	***0.0***	-0.776	***0.0***	-0.76	***0.003***	-0.288	***0.0***	-0.288	*-0.047*	-0.515	***-0.003***	-0.516	***-0.0***	-0.536	***-0.0***	-0.534
	ΔF1	***0.0***	-0.745	***0.0***	-0.731	***0.002***	-0.312	***0.0***	-0.312	*-0.066*	-0.48	***-0.005***	-0.479	***-0.0***	-0.451	***-0.0***	-0.449
	ΔAc	***0.0***	-0.776	***0.0***	-0.76	***0.003***	-0.288	***0.0***	-0.288	*-0.047*	-0.515	***-0.003***	-0.516	***-0.0***	-0.536	***-0.0***	-0.534

Table E.5: Mean of explanation effectiveness of necessary and sufficient explanations with class change condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using RandomForest.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$
Necessary																	
TransH	ΔPr	*-0.126*	0.017	***-0.338***	-0.019	0.045	0.088	0.034	-0.043	-0.07	-0.016	***-0.178***	-0.027	-0.002	0.006	-0.007	-0.002
	ΔRe	***-0.217***	0.006	***-0.371***	-0.005	0.024	0.009	0.003	-0.021	***-0.11***	-0.027	***-0.228***	-0.031	***-0.01***	0.0	***-0.016***	-0.001
	ΔF1	***-0.204***	0.007	***-0.388***	-0.016	0.057	0.011	0.072	-0.031	**-0.083**	-0.028	***-0.183***	-0.033	***-0.008***	0.0	***-0.013***	-0.001
	ΔAc	***-0.217***	0.006	***-0.371***	-0.005	0.024	0.009	0.003	-0.021	***-0.11***	-0.027	***-0.228***	-0.031	***-0.01***	0.0	***-0.016***	-0.001
TransE	ΔPr	***-0.195***	-0.024	***-0.342***	-0.022	-0.032	0.025	-0.034	0.018	***-0.165***	-0.043	***-0.301***	-0.071	0.005	0.008	-0.009	0.0
	ΔRe	***-0.318***	-0.023	***-0.443***	-0.034	-0.015	0.012	-0.021	0.009	***-0.15***	-0.033	***-0.25***	-0.033	***-0.007***	0.001	***-0.013***	0.001
	ΔF1	***-0.327***	-0.021	***-0.466***	-0.033	0.011	0.02	0.022	0.013	***-0.134***	-0.036	***-0.238***	-0.045	***-0.006***	0.001	***-0.01***	0.001
	ΔAc	***-0.318***	-0.023	***-0.443***	-0.034	-0.015	0.012	-0.021	0.009	***-0.15***	-0.033	***-0.25***	-0.033	***-0.007***	0.001	***-0.013***	0.001
Sufficient																	
TransH	ΔPr	***-0.024***	-0.527	***-0.024***	-0.493	**0.004**	0.035	**0.0**	0.019	*-0.078*	-0.249	*-0.078*	-0.252	*-0.01*	-0.079	***-0.002***	-0.088
	ΔRe	***-0.011***	-0.507	***-0.011***	-0.485	**0.003**	-0.003	**0.0**	-0.015	*-0.047*	-0.331	*-0.046*	-0.346	*-0.001*	-0.159	***0.0***	-0.146
	ΔF1	***-0.018***	-0.523	***-0.018***	-0.499	**0.002**	0.076	**0.0**	0.063	*-0.062*	-0.288	*-0.061*	-0.3	***-0.001***	-0.108	***0.0***	-0.104
	ΔAc	***-0.011***	-0.507	***-0.011***	-0.485	**0.003**	-0.003	**0.0**	-0.015	*-0.047*	-0.331	*-0.046*	-0.346	*-0.001*	-0.159	***0.0***	-0.146
TransE	ΔPr	***0.002***	-0.484	***0.002***	-0.583	**0.006**	-0.082	**0.006**	-0.112	*-0.135*	-0.372	*-0.128*	-0.325	***-0.009***	-0.068	**-0.008**	-0.068
	ΔRe	***0.0***	-0.533	***0.0***	-0.528	**0.003**	-0.076	***0.003***	-0.074	*-0.061*	-0.387	*-0.06*	-0.353	***-0.0***	-0.147	***-0.0***	-0.132
	ΔF1	***0.001***	-0.565	***0.001***	-0.566	**0.002**	-0.034	**0.002**	-0.042	*-0.085*	-0.367	*-0.083*	-0.328	***-0.0***	-0.1	***-0.0***	-0.095
	ΔAc	***0.0***	-0.533	***0.0***	-0.528	**0.003**	-0.076	***0.003***	-0.074	*-0.061*	-0.387	*-0.06*	-0.353	***-0.0***	-0.147	***-0.0***	-0.132

Results for All the KGE Methods and XGBoost

Table E.6: Mean of explanation effectiveness of necessary explanations with class change condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using XGBoost.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$
RDF2Vec	ΔPr	***-0.14***	0.0	***-0.362***	0.0	***-0.078***	0.0	***-0.257***	0.0	***-0.335***	0.0	***-0.553***	-0.003	***-0.043***	0.0	***-0.122***	-0.004
	ΔRe	***-0.241***	0.0	***-0.445***	0.0	***-0.1***	0.0	***-0.303***	0.0	***-0.353***	0.0	***-0.612***	-0.005	***-0.034***	0.0	***-0.215***	-0.002
	ΔF1	***-0.245***	0.0	***-0.475***	0.0	***-0.082***	0.0	***-0.318***	0.0	***-0.351***	0.0	***-0.593***	-0.005	***-0.017***	0.0	***-0.132***	-0.001
	ΔAc	***-0.241***	0.0	***-0.445***	0.0	***-0.1***	0.0	***-0.303***	0.0	***-0.353***	0.0	***-0.612***	-0.005	***-0.034***	0.0	***-0.215***	-0.002
ComplEx	ΔPr	***-0.18***	0.0	***-0.477***	0.0	-0.037	0.0	***-0.215***	-0.003	***-0.282***	0.0	***-0.533***	-0.003	***-0.043***	0.0	***-0.152***	-0.002
	ΔRe	***-0.245***	0.0	***-0.484***	0.0	-0.038	0.0	***-0.244***	-0.003	***-0.238***	0.0	***-0.541***	-0.004	***-0.043***	0.0	***-0.243***	-0.002
	ΔF1	***-0.265***	0.0	***-0.506***	0.0	-0.026	0.0	***-0.232***	-0.003	***-0.269***	0.0	***-0.53***	-0.003	***-0.025***	0.0	***-0.157***	-0.001
	ΔAc	***-0.245***	0.0	***-0.484***	0.0	-0.038	0.0	***-0.244***	-0.003	***-0.238***	0.0	***-0.541***	-0.004	***-0.043***	0.0	***-0.243***	-0.002
distMult	ΔPr	***-0.173***	0.0	***-0.441***	0.0	-0.031	0.0	***-0.21***	0.0	***-0.293***	0.0	***-0.524***	-0.005	***-0.016***	0.0	***-0.121***	-0.004
	ΔRe	***-0.234***	0.0	***-0.455***	0.0	-0.026	0.0	***-0.235***	0.0	***-0.262***	0.0	***-0.527***	-0.004	***-0.022***	0.0	***-0.207***	-0.003
	ΔF1	***-0.244***	0.0	***-0.484***	0.0	-0.018	0.0	***-0.228***	0.0	***-0.284***	0.0	***-0.518***	-0.004	-0.006	0.0	***-0.133***	-0.002
	ΔAc	***-0.234***	0.0	***-0.455***	0.0	-0.026	0.0	***-0.235***	0.0	***-0.262***	0.0	***-0.527***	-0.004	***-0.022***	0.0	***-0.207***	-0.003
TransH	ΔPr	***-0.157***	0.0	***-0.361***	0.0	-0.072	0.0	***-0.131***	0.007	***-0.222***	0.0	***-0.505***	-0.006	-0.014	0.0	***-0.095***	-0.001
	ΔRe	***-0.159***	0.0	***-0.386***	0.0	-0.076	0.0	***-0.171***	0.006	***-0.216***	0.0	***-0.489***	-0.003	***-0.043***	0.0	***-0.264***	-0.001
	ΔF1	***-0.18***	0.0	***-0.388***	0.0	-0.062	0.0	***-0.15***	0.007	***-0.202***	0.0	***-0.484***	-0.003	***-0.014***	0.0	***-0.154***	0.0
	ΔAc	***-0.159***	0.0	***-0.386***	0.0	-0.076	0.0	***-0.171***	0.006	***-0.216***	0.0	***-0.489***	-0.003	***-0.043***	0.0	***-0.264***	-0.001
TransE	ΔPr	***-0.125***	0.0	***-0.418***	0.0	-0.007	0.0	-0.099	0.006	***-0.197***	0.0	***-0.481***	-0.002	***-0.027***	0.0	***-0.114***	-0.003
	ΔRe	***-0.267***	0.0	***-0.516***	0.0	-0.018	0.0	***-0.138***	0.006	***-0.24***	0.0	***-0.496***	-0.001	***-0.045***	0.0	***-0.28***	-0.001
	ΔF1	***-0.257***	0.0	***-0.532***	0.0	-0.016	0.0	-0.128	0.007	***-0.215***	0.0	***-0.483***	-0.001	-0.011	0.0	***-0.167***	-0.0
	ΔAc	***-0.267***	0.0	***-0.516***	0.0	-0.018	0.0	***-0.138***	0.006	***-0.24***	0.0	***-0.496***	-0.001	***-0.045***	0.0	***-0.28***	-0.001

Table E.7: Mean of explanation effectiveness of sufficient explanations with class change condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using XGBoost.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$
RDF2Vec	ΔPr	***-0.009***	-0.919	***0.0***	-0.89	***0.0***	-0.38	***0.0***	-0.38	*-0.021*	-0.78	***0.001***	-0.744	*-0.034*	-0.432	***-0.002***	-0.422
	ΔRe	***-0.011***	-0.909	***0.0***	-0.898	***0.0***	-0.406	***0.0***	-0.406	*-0.022*	-0.777	***0.0***	-0.756	*-0.006*	-0.546	***-0.0***	-0.543
	ΔF1	***-0.014***	-0.91	***0.0***	-0.895	***0.0***	-0.41	***0.0***	-0.41	*-0.022*	-0.777	***0.001***	-0.755	*-0.015*	-0.488	***-0.001***	-0.482
	ΔAc	***-0.011***	-0.909	***0.0***	-0.898	***0.0***	-0.406	***0.0***	-0.406	*-0.022*	-0.777	***0.0***	-0.756	*-0.006*	-0.546	***-0.0***	-0.543
ComplEx	ΔPr	***0.0***	-0.761	***0.0***	-0.776	***0.0***	-0.432	***0.0***	-0.432	*-0.133*	-0.694	***0.002***	-0.658	***-0.012***	-0.437	***-0.0***	-0.433
	ΔRe	***0.0***	-0.76	***0.0***	-0.777	***0.0***	-0.441	***0.0***	-0.441	*-0.11*	-0.69	***0.002***	-0.679	*-0.004*	-0.556	***-0.0***	-0.553
	ΔF1	***0.0***	-0.751	***0.0***	-0.767	***0.0***	-0.446	***0.0***	-0.446	*-0.131*	-0.683	***0.002***	-0.667	*-0.012*	-0.482	***-0.0***	-0.479
	ΔAc	***0.0***	-0.76	***0.0***	-0.777	***0.0***	-0.441	***0.0***	-0.441	*-0.11*	-0.69	***0.002***	-0.679	*-0.004*	-0.556	***-0.0***	-0.553
distMult	ΔPr	***0.0***	-0.815	***0.0***	-0.828	***0.0***	-0.411	***0.0***	-0.411	*-0.171*	-0.661	***-0.002***	-0.66	***-0.003***	-0.453	***-0.001***	-0.452
	ΔRe	***0.0***	-0.817	***0.0***	-0.817	***0.0***	-0.394	***0.0***	-0.394	*-0.13*	-0.632	***0.001***	-0.626	***-0.002***	-0.578	***-0.0***	-0.576
	ΔF1	***-0.001***	-0.81	***0.0***	-0.81	***0.0***	-0.404	***0.0***	-0.404	*-0.159*	-0.643	***0.001***	-0.634	*-0.004*	-0.516	***-0.0***	-0.515
	ΔAc	***0.0***	-0.817	***0.0***	-0.817	***0.0***	-0.394	***0.0***	-0.394	*-0.13*	-0.632	***0.001***	-0.626	***-0.002***	-0.578	***-0.0***	-0.576
TransH	ΔPr	***0.009***	-0.727	***0.0***	-0.732	***-0.003***	-0.218	***0.0***	-0.218	*-0.078*	-0.571	***0.0***	-0.603	*-0.024*	-0.384	***0.002***	-0.385
	ΔRe	***0.005***	-0.675	***0.0***	-0.664	***0.009***	-0.259	***0.0***	-0.259	*-0.039*	-0.567	***0.0***	-0.588	*-0.003*	-0.544	***-0.0***	-0.548
	ΔF1	***-0.001***	-0.696	***0.0***	-0.694	***-0.046***	-0.266	***0.0***	-0.266	*-0.054*	-0.56	***0.0***	-0.578	*-0.007*	-0.468	***-0.0***	-0.471
	ΔAc	***0.005***	-0.675	***0.0***	-0.664	***0.009***	-0.259	***0.0***	-0.259	*-0.039*	-0.567	***0.0***	-0.588	*-0.003*	-0.544	***-0.0***	-0.548
TransE	ΔPr	***0.0***	-0.739	***0.0***	-0.761	***0.043***	-0.252	***0.0***	-0.248	*-0.057*	-0.587	***0.0***	-0.587	*-0.025*	-0.421	***-0.002***	-0.425
	ΔRe	***-0.006***	-0.77	***0.0***	-0.782	***0.026***	-0.276	***0.0***	-0.274	*-0.051*	-0.594	***-0.001***	-0.59	***-0.002***	-0.544	***-0.0***	-0.545
	ΔF1	***-0.005***	-0.754	***0.0***	-0.769	***-0.005***	-0.277	***0.0***	-0.275	*-0.063*	-0.577	***-0.001***	-0.575	*-0.006*	-0.473	***-0.001***	-0.475
	ΔAc	***-0.006***	-0.77	***0.0***	-0.782	***0.026***	-0.276	***0.0***	-0.274	*-0.051*	-0.594	***-0.001***	-0.59	***-0.002***	-0.544	***-0.0***	-0.545

Table E.8: Mean of explanation effectiveness of necessary explanations with class probability condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using XGBoost.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$
RDF2Vec	ΔPr	***-0.14***	0.0	***-0.219***	-0.006	***-0.078***	0.003	***-0.082***	-0.011	***-0.332***	-0.003	***-0.382***	-0.01	***-0.043***	-0.005	***-0.065***	-0.003
	ΔRe	***-0.241***	0.0	***-0.32***	-0.006	***-0.1***	0.0	***-0.121***	-0.012	***-0.351***	-0.002	***-0.411***	-0.007	***-0.034***	0.001	***-0.07***	0.0
	ΔF1	***-0.245***	0.0	***-0.333***	-0.006	***-0.082***	-0.001	***-0.104***	-0.011	***-0.349***	-0.002	***-0.403***	-0.007	***-0.017***	-0.0	***-0.039***	-0.0
	ΔAc	***-0.241***	0.0	***-0.32***	-0.006	***-0.1***	0.0	***-0.121***	-0.012	***-0.351***	-0.002	***-0.411***	-0.007	***-0.034***	0.001	***-0.07***	0.0
ComplEx	ΔPr	***-0.18***	0.003	***-0.232***	-0.006	-0.037	-0.016	-0.029	-0.01	***-0.281***	0.001	***-0.345***	0.001	***-0.043***	0.006	***-0.062***	0.003
	ΔRe	***-0.245***	0.006	***-0.285***	-0.006	-0.038	-0.015	-0.035	-0.009	***-0.237***	-0.002	***-0.322***	-0.005	***-0.043***	0.001	***-0.074***	0.001
	ΔF1	***-0.265***	0.006	***-0.308***	-0.006	-0.026	-0.014	-0.024	-0.009	***-0.268***	-0.003	***-0.339***	-0.003	***-0.026***	0.001	***-0.047***	0.002
	ΔAc	***-0.245***	0.006	***-0.285***	-0.006	-0.038	-0.015	-0.035	-0.009	***-0.237***	-0.002	***-0.322***	-0.005	***-0.043***	0.001	***-0.074***	0.001
distMult	ΔPr	***-0.173***	0.005	***-0.203***	0.01	-0.031	0.001	-0.039	0.007	***-0.293***	0.0	***-0.352***	-0.006	**-0.015**	0.005	***-0.032***	0.007
	ΔRe	***-0.234***	0.006	***-0.268***	0.011	-0.026	0.0	-0.035	0.006	***-0.263***	0.005	***-0.342***	0.003	***-0.022***	0.002	***-0.049***	0.001
	ΔF1	***-0.244***	0.006	***-0.287***	0.014	-0.018	0.001	-0.026	0.006	***-0.285***	0.001	***-0.346***	0.0	-0.007	0.004	***-0.024***	0.002
	ΔAc	***-0.234***	0.006	***-0.268***	0.011	-0.026	0.0	-0.035	0.006	***-0.263***	0.005	***-0.342***	0.003	***-0.022***	0.002	***-0.049***	0.001
TransH	ΔPr	***-0.157***	-0.011	***-0.166***	-0.017	-0.072	0.01	-0.072	-0.004	***-0.226***	-0.014	***-0.284***	-0.017	-0.014	0.007	-0.019	0.004
	ΔRe	***-0.159***	-0.023	***-0.182***	-0.017	*-0.076*	0.009	-0.085	-0.006	***-0.219***	0.002	***-0.301***	-0.001	***-0.042***	0.001	***-0.067***	0.002
	ΔF1	***-0.18***	-0.024	***-0.199***	-0.023	-0.062	0.009	-0.066	-0.004	***-0.205***	0.001	***-0.277***	-0.004	***-0.014***	0.001	***-0.024***	0.002
	ΔAc	***-0.159***	-0.023	***-0.182***	-0.017	*-0.076*	0.009	-0.085	-0.006	***-0.219***	0.002	***-0.301***	-0.001	***-0.042***	0.001	***-0.067***	0.002
TransE	ΔPr	***-0.125***	0.003	***-0.165***	-0.009	-0.007	0.007	-0.011	-0.006	***-0.194***	-0.007	***-0.263***	0.003	***-0.028***	-0.009	***-0.035***	0.003
	ΔRe	***-0.267***	0.0	***-0.307***	-0.017	-0.018	0.012	-0.021	-0.003	***-0.237***	0.003	***-0.317***	0.003	***-0.045***	-0.003	***-0.068***	0.001
	ΔF1	***-0.257***	0.0	***-0.304***	-0.015	-0.016	0.011	-0.016	-0.005	***-0.211***	0.001	***-0.288***	0.004	-0.012	-0.003	***-0.022***	0.001
	ΔAc	***-0.267***	0.0	***-0.307***	-0.017	-0.018	0.012	-0.021	-0.003	***-0.237***	0.003	***-0.317***	0.003	***-0.045***	-0.003	***-0.068***	0.001

Table E.9: Mean of explanation effectiveness of sufficient explanations with class probability condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using XGBoost.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$
RDF2Vec	ΔPr	***-0.009***	-0.544	***0.0***	-0.516	***0.0***	-0.234	***0.0***	-0.21	*-0.024*	-0.551	***-0.0***	-0.626	*-0.035*	-0.211	***-0.008***	-0.208
	ΔRe	***-0.011***	-0.567	***0.0***	-0.526	***0.0***	-0.238	***0.0***	-0.221	*-0.026*	-0.617	***-0.002***	-0.64	*-0.006*	-0.366	***-0.001***	-0.361
	ΔF1	***-0.014***	-0.584	***0.0***	-0.544	***0.0***	-0.225	***0.0***	-0.206	*-0.025*	-0.606	***-0.001***	-0.635	*-0.015*	-0.282	*-0.004*	-0.278
	ΔAc	***-0.011***	-0.567	***0.0***	-0.526	***0.0***	-0.238	***0.0***	-0.221	*-0.026*	-0.617	***-0.002***	-0.64	*-0.006*	-0.366	***-0.001***	-0.361
ComplEx	ΔPr	***0.0***	-0.584	***0.0***	-0.505	***0.0***	-0.247	***0.0***	-0.301	*-0.135*	-0.571	***-0.005***	-0.53	***-0.011***	-0.203	***0.002***	-0.195
	ΔRe	***0.0***	-0.563	***0.0***	-0.546	***0.0***	-0.279	***0.0***	-0.335	*-0.113*	-0.551	***-0.006***	-0.532	*-0.004*	-0.32	***0.0***	-0.309
	ΔF1	***0.0***	-0.578	***0.0***	-0.557	***0.0***	-0.271	***0.0***	-0.326	*-0.134*	-0.56	***-0.006***	-0.533	*-0.013*	-0.233	***-0.001***	-0.222
	ΔAc	***0.0***	-0.563	***0.0***	-0.546	***0.0***	-0.279	***0.0***	-0.335	*-0.113*	-0.551	***-0.006***	-0.532	*-0.004*	-0.32	***0.0***	-0.309
distMult	ΔPr	***0.0***	-0.6	***0.004***	-0.529	***0.0***	-0.248	***0.0***	-0.248	*-0.172*	-0.597	***-0.01***	-0.553	*-0.005*	-0.173	***-0.001***	-0.175
	ΔRe	***0.0***	-0.642	***0.006***	-0.601	***0.0***	-0.25	***0.0***	-0.247	*-0.131*	-0.532	***-0.008***	-0.512	*-0.002*	-0.328	***-0.0***	-0.329
	ΔF1	***-0.001***	-0.627	***0.006***	-0.585	***0.0***	-0.239	***0.0***	-0.234	*-0.16*	-0.57	***-0.008***	-0.542	*-0.005*	-0.236	***-0.001***	-0.237
	ΔAc	***0.0***	-0.642	***0.006***	-0.601	***0.0***	-0.25	***0.0***	-0.247	*-0.131*	-0.532	***-0.008***	-0.512	*-0.002*	-0.328	***-0.0***	-0.329
TransH	ΔPr	***0.009***	-0.475	***-0.009***	-0.384	***-0.003***	-0.172	***-0.006***	-0.176	*-0.085*	-0.516	***-0.013***	-0.458	*-0.025*	-0.154	***-0.006***	-0.164
	ΔRe	***0.005***	-0.432	***-0.011***	-0.399	***0.009***	-0.179	***-0.009***	-0.176	*-0.043*	-0.529	***-0.009***	-0.508	*-0.003*	-0.37	***-0.001***	-0.371
	ΔF1	***-0.001***	-0.463	***-0.014***	-0.421	***-0.046***	-0.164	***-0.012***	-0.165	*-0.059*	-0.512	*-0.014*	-0.48	*-0.007*	-0.261	*-0.003*	-0.263
	ΔAc	***0.005***	-0.432	***-0.011***	-0.399	***0.009***	-0.179	***-0.009***	-0.176	*-0.043*	-0.529	***-0.009***	-0.508	*-0.003*	-0.37	***-0.001***	-0.371
TransE	ΔPr	***0.0***	-0.472	***0.0***	-0.555	***0.043***	-0.152	***-0.002***	-0.139	*-0.068*	-0.449	*-0.029*	-0.46	*-0.026*	-0.167	***-0.005***	-0.175
	ΔRe	***-0.006***	-0.5	***-0.006***	-0.573	***0.026***	-0.147	**0.0**	-0.141	*-0.056*	-0.485	*-0.014*	-0.502	*-0.003*	-0.396	***-0.001***	-0.39
	ΔF1	***-0.005***	-0.504	***-0.005***	-0.583	***-0.005***	-0.157	***-0.003***	-0.146	*-0.071*	-0.461	*-0.019*	-0.476	*-0.006*	-0.287	***-0.002***	-0.284
	ΔAc	***-0.006***	-0.5	***-0.006***	-0.573	***0.026***	-0.147	**0.0**	-0.141	*-0.056*	-0.485	*-0.014*	-0.502	*-0.003*	-0.396	***-0.001***	-0.39

Results for All the KGE Methods and MLP

Table E.10: Mean of explanation effectiveness of necessary explanations with class change condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using MLP.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$	single	$rand_s$	comp	$rand_c$
RDF2Vec	ΔPr	0.009	0.0	***-0.157***	0.0	***-0.103***	0.0	***-0.167***	0.0	***-0.326***	0.0	***-0.598***	0.001	***-0.026***	0.0	***-0.117***	-0.001
	ΔRe	-0.017	0.0	***-0.21***	0.0	***-0.179***	0.0	***-0.215***	0.0	***-0.296***	0.0	***-0.633***	0.001	***-0.029***	0.0	***-0.231***	-0.001
	ΔF1	-0.01	0.0	***-0.203***	0.0	***-0.189***	0.0	***-0.238***	0.0	***-0.321***	0.0	***-0.625***	0.001	***-0.023***	0.0	***-0.163***	-0.001
	ΔAc	-0.017	0.0	***-0.21***	0.0	***-0.179***	0.0	***-0.215***	0.0	***-0.296***	0.0	***-0.633***	0.001	***-0.029***	0.0	***-0.231***	-0.001
ComplEx	ΔPr	***-0.092***	0.0	***-0.404***	0.0	-0.013	0.0	-0.061	0.0	***-0.091***	0.0	***-0.416***	-0.002	***-0.035***	0.0	***-0.145***	-0.002
	ΔRe	***-0.178***	0.0	***-0.411***	0.0	-0.015	0.0	-0.074	0.0	***-0.123***	0.0	***-0.452***	-0.001	***-0.045***	0.0	***-0.275***	-0.002
	ΔF1	***-0.181***	0.0	***-0.432***	0.0	-0.013	0.0	-0.065	0.0	***-0.107***	0.0	***-0.451***	-0.002	***-0.032***	0.0	***-0.199***	-0.001
	ΔAc	***-0.178***	0.0	***-0.411***	0.0	-0.015	0.0	-0.074	0.0	***-0.123***	0.0	***-0.452***	-0.001	***-0.045***	0.0	***-0.275***	-0.002
distMult	ΔPr	***-0.172***	0.0	***-0.393***	0.0	0.005	0.0	-0.099	0.0	***-0.161***	0.0	***-0.444***	-0.0	***-0.022***	0.0	***-0.134***	-0.001
	ΔRe	***-0.24***	0.0	***-0.45***	0.0	-0.006	0.0	***-0.147***	0.0	***-0.185***	0.0	***-0.489***	0.0	***-0.028***	0.0	***-0.261***	-0.001
	ΔF1	***-0.247***	0.0	***-0.47***	0.0	-0.002	0.0	***-0.146***	0.0	***-0.174***	0.0	***-0.482***	-0.0	***-0.022***	0.0	***-0.194***	-0.001
	ΔAc	***-0.24***	0.0	***-0.45***	0.0	-0.006	0.0	***-0.147***	0.0	***-0.185***	0.0	***-0.489***	0.0	***-0.028***	0.0	***-0.261***	-0.001
TransH	ΔPr	-0.116	0.0	***-0.339***	0.0	***-0.053***	0.0	***-0.109***	0.0	-0.016	0.0	***-0.258***	0.0	-0.003	0.0	***-0.089***	-0.0
	ΔRe	***-0.246***	0.0	***-0.495***	0.0	***-0.05***	0.0	***-0.121***	0.0	***-0.091***	0.0	***-0.328***	0.0	***-0.028***	0.0	***-0.191***	-0.0
	ΔF1	***-0.201***	0.0	***-0.436***	0.0	***-0.041***	0.0	***-0.106***	0.0	***-0.048***	0.0	***-0.31***	0.0	***-0.019***	0.0	***-0.128***	-0.0
	ΔAc	***-0.246***	0.0	***-0.495***	0.0	***-0.05***	0.0	***-0.121***	0.0	***-0.091***	0.0	***-0.328***	0.0	***-0.028***	0.0	***-0.191***	-0.0
TransE	ΔPr	-0.112	0.0	***-0.339***	0.0	0.0	0.0	-0.081	0.0	***-0.041***	0.0	***-0.231***	0.0	0.015	0.0	***-0.085***	-0.0
	ΔRe	***-0.247***	0.0	***-0.496***	0.0	-0.015	0.0	-0.126	0.0	***-0.091***	0.0	***-0.326***	0.0	***-0.025***	0.0	***-0.199***	-0.0
	ΔF1	***-0.198***	0.0	***-0.449***	0.0	-0.006	0.0	-0.12	0.0	***-0.059***	0.0	***-0.289***	0.0	***-0.014***	0.0	***-0.125***	-0.0
	ΔAc	***-0.247***	0.0	***-0.496***	0.0	-0.015	0.0	-0.126	0.0	***-0.091***	0.0	***-0.326***	0.0	***-0.025***	0.0	***-0.199***	-0.0

Table E.11: Mean of explanation effectiveness of sufficient explanations with class change condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using MLP.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$
RDF2Vec	ΔPr	***0.0***	-0.718	***0.0***	-0.721	***0.0***	-0.269	***0.0***	-0.269	***-0.0***	-0.797	***0.001***	-0.824	*-0.013*	-0.394	***0.0***	-0.386
	ΔRe	***0.0***	-0.663	***0.0***	-0.657	***0.0***	-0.282	***0.0***	-0.282	***-0.003***	-0.808	***0.001***	-0.822	*-0.006*	-0.541	***0.0***	-0.536
	ΔF1	***0.0***	-0.708	***0.0***	-0.702	***0.0***	-0.286	***0.0***	-0.286	***-0.002***	-0.804	***0.001***	-0.821	*-0.01*	-0.497	***0.0***	-0.491
	ΔAc	***0.0***	-0.663	***0.0***	-0.657	***0.0***	-0.282	***0.0***	-0.282	***-0.003***	-0.808	***0.001***	-0.822	*-0.006*	-0.541	***0.0***	-0.536
ComplEx	ΔPr	***0.001***	-0.803	***0.0***	-0.824	***0.0***	-0.269	***0.0***	-0.269	*-0.076*	-0.57	***0.0***	-0.559	***-0.002***	-0.372	***0.0***	-0.382
	ΔRe	***0.0***	-0.741	***0.0***	-0.757	***0.0***	-0.294	***0.0***	-0.294	*-0.042*	-0.579	***0.0***	-0.567	***-0.002***	-0.547	***0.0***	-0.55
	ΔF1	***-0.0***	-0.75	***0.0***	-0.769	***0.0***	-0.296	***0.0***	-0.296	*-0.06*	-0.572	***0.0***	-0.56	***-0.001***	-0.502	***0.0***	-0.504
	ΔAc	***0.0***	-0.741	***0.0***	-0.757	***0.0***	-0.294	***0.0***	-0.294	*-0.042*	-0.579	***0.0***	-0.567	***-0.002***	-0.547	***0.0***	-0.55
distMult	ΔPr	***0.0***	-0.908	***0.0***	-0.922	***0.0***	-0.306	***0.0***	-0.306	***-0.008***	-0.574	***0.0***	-0.588	***-0.004***	-0.399	***-0.0***	-0.357
	ΔRe	***0.0***	-0.886	***0.0***	-0.897	***0.0***	-0.329	***0.0***	-0.329	***-0.008***	-0.595	***0.0***	-0.603	*-0.019*	-0.575	***0.0***	-0.568
	ΔF1	***0.0***	-0.888	***0.0***	-0.901	***0.0***	-0.33	***0.0***	-0.33	***-0.009***	-0.593	***0.0***	-0.601	*-0.009*	-0.54	***-0.0***	-0.529
	ΔAc	***0.0***	-0.886	***0.0***	-0.897	***0.0***	-0.329	***0.0***	-0.329	***-0.008***	-0.595	***0.0***	-0.603	*-0.019*	-0.575	***0.0***	-0.568
TransH	ΔPr	***-0.028***	-0.815	***0.0***	-0.789	***0.0***	-0.149	***0.0***	-0.149	***-0.011***	-0.32	***-0.011***	-0.306	*-0.005*	-0.338	***-0.003***	-0.346
	ΔRe	***-0.017***	-0.84	***0.0***	-0.828	***0.0***	-0.171	***0.0***	-0.171	***-0.004***	-0.418	***-0.004***	-0.404	***-0.001***	-0.57	***-0.0***	-0.571
	ΔF1	***-0.022***	-0.825	***0.0***	-0.812	***0.0***	-0.175	***0.0***	-0.175	*-0.016*	-0.356	***-0.006***	-0.344	*-0.001*	-0.468	***-0.0***	-0.469
	ΔAc	***-0.017***	-0.84	***0.0***	-0.828	***0.0***	-0.171	***0.0***	-0.171	***-0.004***	-0.418	***-0.004***	-0.404	***-0.001***	-0.57	***-0.0***	-0.571
TransE	ΔPr	***0.0***	-0.869	***0.0***	-0.86	***0.0***	-0.218	***0.0***	-0.218	*-0.026*	-0.268	*-0.004*	-0.253	***0.0***	-0.363	***0.0***	-0.357
	ΔRe	***0.0***	-0.858	***0.0***	-0.852	***0.0***	-0.25	***0.0***	-0.25	***-0.003***	-0.389	***-0.002***	-0.383	***0.0***	-0.574	***0.0***	-0.572
	ΔF1	***0.0***	-0.851	***0.0***	-0.844	***0.0***	-0.251	***0.0***	-0.251	*-0.021*	-0.32	*-0.004*	-0.311	***0.0***	-0.475	***0.0***	-0.472
	ΔAc	***0.0***	-0.858	***0.0***	-0.852	***0.0***	-0.25	***0.0***	-0.25	***-0.003***	-0.389	***-0.002***	-0.383	***0.0***	-0.574	***0.0***	-0.572

Table E.12: Mean of explanation effectiveness of necessary explanations with class probability condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using MLP.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$
RDF2Vec	ΔPr	0.009	0.0	-0.043	0.0	***-0.103***	0.001	***-0.12***	0.004	***-0.326***	-0.0	***-0.365***	-0.001	***-0.024***	-0.0	***-0.047***	-0.0
	ΔRe	-0.017	0.0	-0.062	0.0	***-0.179***	-0.0	***-0.174***	-0.0	***-0.297***	-0.002	***-0.36***	-0.0	***-0.028***	-0.002	***-0.079***	-0.001
	ΔF1	-0.01	0.0	-0.056	0.0	***-0.189***	0.0	***-0.171***	-0.002	***-0.323***	-0.001	***-0.375***	-0.0	***-0.023***	-0.001	***-0.053***	-0.0
	ΔAc	-0.017	0.0	-0.062	0.0	***-0.179***	-0.0	***-0.174***	-0.0	***-0.297***	-0.002	***-0.36***	-0.0	***-0.028***	-0.002	***-0.079***	-0.001
ComplEx	ΔPr	***-0.092***	0.006	***-0.14***	0.006	-0.013	0.002	0.002	0.005	***-0.089***	-0.017	***-0.136***	0.001	***-0.034***	-0.0	***-0.056***	-0.0
	ΔRe	***-0.178***	0.0	***-0.224***	0.0	-0.015	0.0	0.003	0.006	***-0.121***	-0.01	***-0.191***	0.001	***-0.043***	-0.001	***-0.094***	-0.002
	ΔF1	***-0.181***	0.003	***-0.229***	0.003	-0.013	-0.0	0.008	0.006	***-0.105***	-0.012	***-0.167***	0.0	***-0.031***	-0.001	***-0.061***	-0.001
	ΔAc	***-0.178***	0.0	***-0.224***	0.0	-0.015	0.0	0.003	0.006	***-0.121***	-0.01	***-0.191***	0.001	***-0.043***	-0.001	***-0.094***	-0.002
distMult	ΔPr	***-0.172***	0.0	***-0.25***	0.003	0.005	-0.003	-0.022	0.003	***-0.158***	0.003	***-0.206***	-0.009	***-0.021***	0.004	***-0.037***	-0.001
	ΔRe	***-0.24***	0.0	***-0.302***	0.0	-0.006	-0.003	-0.035	0.003	***-0.182***	0.003	***-0.257***	-0.004	***-0.027***	0.001	***-0.067***	-0.001
	ΔF1	***-0.247***	0.0	***-0.317***	0.002	-0.002	-0.004	-0.031	0.003	***-0.17***	-0.001	***-0.237***	-0.006	***-0.021***	0.003	***-0.044***	-0.001
	ΔAc	***-0.24***	0.0	***-0.302***	0.0	-0.006	-0.003	-0.035	0.003	***-0.182***	0.003	***-0.257***	-0.004	***-0.027***	0.001	***-0.067***	-0.001
TransH	ΔPr	-0.116	-0.031	-0.142	-0.027	***-0.053***	0.001	***-0.052***	-0.0	-0.012	-0.025	***-0.182***	-0.074	-0.004	0.007	-0.003	-0.004
	ΔRe	***-0.246***	-0.017	***-0.274***	-0.011	***-0.05***	0.0	***-0.062***	0.0	***-0.088***	-0.017	***-0.224***	-0.006	***-0.027***	-0.003	***-0.043***	-0.002
	ΔF1	***-0.201***	-0.027	***-0.224***	-0.021	**-0.041**	0.003	-0.049	0.002	**-0.045**	-0.021	***-0.194***	-0.059	***-0.019***	-0.002	***-0.022***	-0.002
	ΔAc	***-0.246***	-0.017	***-0.274***	-0.011	***-0.05***	0.0	***-0.062***	0.0	***-0.088***	-0.017	***-0.224***	-0.006	***-0.027***	-0.003	***-0.043***	-0.002
TransE	ΔPr	-0.106	-0.027	**-0.117**	-0.015	0.0	-0.011	-0.023	0.005	***-0.04***	0.004	***-0.159***	-0.053	0.01	-0.004	-0.0	-0.002
	ΔRe	***-0.241***	-0.011	***-0.263***	-0.006	-0.015	-0.012	-0.035	0.006	***-0.094***	0.005	***-0.186***	0.004	***-0.024***	-0.003	***-0.051***	-0.002
	ΔF1	***-0.193***	-0.016	***-0.208***	-0.01	-0.006	-0.012	-0.029	0.006	***-0.06***	0.002	***-0.168***	-0.038	***-0.014***	-0.003	***-0.025***	-0.001
	ΔAc	***-0.241***	-0.011	***-0.263***	-0.006	-0.015	-0.012	-0.035	0.006	***-0.094***	0.005	***-0.186***	0.004	***-0.024***	-0.003	***-0.051***	-0.002

Table E.13: Mean of explanation effectiveness of sufficient explanations with class probability condition, measured based on the precision (Pr), recall (Re) and weighted average F1-score (F1) and accuracy (Ac) variation for predictions using MLP.

		AIFB				**MUTAG**				**AM**				**MDGENRE**			
		single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$	single	rand$_s$	comp	rand$_c$
RDF2Vec	ΔPr	***0.0***	-0.545	***0.0***	-0.492	***0.0***	-0.239	***0.0***	-0.216	***-0.0***	-0.631	***-0.0***	-0.63	*-0.013*	-0.216	*-0.004*	-0.218
	ΔRe	***0.0***	-0.521	***0.0***	-0.51	***0.0***	-0.265	***0.0***	-0.247	***-0.003***	-0.677	***0.0***	-0.665	*-0.007*	-0.456	*-0.002*	-0.45
	ΔF1	***0.0***	-0.577	***0.0***	-0.557	***0.0***	-0.26	***0.0***	-0.242	***-0.002***	-0.667	***-0.0***	-0.656	*-0.011*	-0.4	*-0.004*	-0.392
	ΔAc	***0.0***	-0.521	***0.0***	-0.51	***0.0***	-0.265	***0.0***	-0.247	***-0.003***	-0.677	***0.0***	-0.665	*-0.007*	-0.456	*-0.002*	-0.45
ComplEx	ΔPr	***0.001***	-0.598	***0.0***	-0.623	***0.0***	-0.233	***0.0***	-0.207	*-0.076*	-0.469	***-0.007***	-0.472	***-0.002***	-0.258	***-0.002***	-0.27
	ΔRe	***0.0***	-0.586	***0.0***	-0.592	***0.0***	-0.259	***0.0***	-0.226	*-0.042*	-0.523	***-0.007***	-0.525	***-0.002***	-0.5	***-0.001***	-0.502
	ΔF1	***-0.0***	-0.583	***0.0***	-0.602	***0.0***	-0.256	***0.0***	-0.219	*-0.06*	-0.505	***-0.007***	-0.507	***-0.002***	-0.433	***-0.001***	-0.44
	ΔAc	***0.0***	-0.586	***0.0***	-0.592	***0.0***	-0.259	***0.0***	-0.226	*-0.042*	-0.523	***-0.007***	-0.525	***-0.002***	-0.5	***-0.001***	-0.502
distMult	ΔPr	***0.0***	-0.658	***0.0***	-0.648	***0.0***	-0.261	***0.0***	-0.278	***-0.008***	-0.521	***-0.005***	-0.527	***-0.004***	-0.177	***0.0***	-0.184
	ΔRe	***0.0***	-0.659	***0.0***	-0.676	***0.0***	-0.288	***0.0***	-0.303	***-0.008***	-0.566	***-0.002***	-0.57	*-0.021*	-0.522	*-0.002*	-0.518
	ΔF1	***0.0***	-0.666	***0.0***	-0.669	***0.0***	-0.284	***0.0***	-0.299	***-0.009***	-0.558	***-0.002***	-0.562	*-0.01*	-0.461	***-0.0***	-0.457
	ΔAc	***0.0***	-0.659	***0.0***	-0.676	***0.0***	-0.288	***0.0***	-0.303	***-0.008***	-0.566	***-0.002***	-0.57	*-0.021*	-0.522	*-0.002*	-0.518
TransH	ΔPr	***-0.028***	-0.617	***-0.028***	-0.612	***0.0***	-0.141	***0.0***	-0.125	***-0.016***	-0.241	***-0.016***	-0.283	*-0.005*	-0.132	***-0.005***	-0.132
	ΔRe	***-0.017***	-0.721	***-0.017***	-0.675	***0.0***	-0.168	***0.0***	-0.156	***-0.001***	-0.345	***-0.001***	-0.352	***-0.001***	-0.441	***-0.0***	-0.441
	ΔF1	***-0.022***	-0.69	***-0.022***	-0.65	***0.0***	-0.163	***0.0***	-0.152	*-0.025*	-0.326	*-0.025*	-0.336	*-0.001*	-0.302	***-0.001***	-0.301
	ΔAc	***-0.017***	-0.721	***-0.017***	-0.675	***0.0***	-0.168	***0.0***	-0.156	***-0.001***	-0.345	***-0.001***	-0.352	***-0.001***	-0.441	***-0.0***	-0.441
TransE	ΔPr	***0.0***	-0.561	***0.0***	-0.497	***0.0***	-0.206	***0.0***	-0.189	*-0.036*	-0.209	*-0.036*	-0.209	***0.0***	-0.14	***-0.001***	-0.143
	ΔRe	***0.0***	-0.672	***0.0***	-0.658	***0.0***	-0.247	***0.0***	-0.226	***0.005***	-0.327	***0.005***	-0.329	***0.0***	-0.447	***0.0***	-0.448
	ΔF1	***0.0***	-0.639	***0.0***	-0.614	***0.0***	-0.244	***0.0***	-0.223	*-0.029*	-0.299	*-0.029*	-0.302	***0.001***	-0.306	***0.0***	-0.305
	ΔAc	***0.0***	-0.672	***0.0***	-0.658	***0.0***	-0.247	***0.0***	-0.226	***0.005***	-0.327	***0.005***	-0.329	***0.0***	-0.447	***0.0***	-0.448