CentraleSupélec

Knowledge Graphs & eXplainable AI (XAI)

Seong Woo AHN
Paul BÉRARD
Charlotte SASSON

Paris
February 2023

Contents

1 Introduction

eXplainable AI (XAI) refers to the development of AI systems that can provide human-understandable explanations for their decisions and actions. This is an important area of research as it addresses the growing concern around the lack of transparency in AI systems, which is a limit to the adoption of AI in industry at scale particularly in domains where the consequences of incorrect decisions can be severe. eXplainable AI aims to improve the trustworthiness and accountability of AI systems by making their decision-making processes more transparent and understandable.

The field of XAI is still in its early stages of development, but it has seen significant progress in recent years. One of the major challenges associated with it is the lack of a clear and agreed upon definition of what constitutes "explanation" in the context of AI systems. As a result, different approaches have been proposed to address this challenge, including:

- Model-based Explanations: These approaches focus on providing an understanding of the inner workings of an AI model, such as by highlighting the input features that have the most influence on a particular prediction.
- Post-hoc Explanations: These approaches generate explanations after the AI system has made its prediction, by identifying the reasoning behind its decision based on the inputs and outputs.
- Concept-based Explanations: These approaches aim to provide explanations in terms of high-level concepts that are meaningful to a human, rather than focusing on the details of the AI model itself.

Despite the progress that has been made in XAI, there are still many challenges that need to be addressed. Some of these challenges include:

- Scalability: Most current XAI methods are computationally expensive and not scalable to large, complex AI models.
- Robustness: XAI explanations can be sensitive to small changes in the data, which can result in different explanations for the same prediction.
- Verifiability: There is currently a lack of methods to verify the accuracy and consistency of XAI explanations, making it difficult to assess their reliability.
- Human Understanding: XAI explanations need to be presented in a way that is easy for humans to understand, which can be challenging given the complexity of AI models.
- Fairness: XAI explanations need to be fair and not perpetuate biases that are present in the data used to train the AI model.

The use of Knowledge Graphs could help overcome the challenges of XAI. A Knowledge Graph is a data structure that represents relationships between entities and their properties in a highly interconnected and structured manner. It enables a meaningful representation of data, making it easier for the decision maker to discover and understand relationships between various pieces of information. This way of representing the data has become increasingly popular in industries such as e-commerce, finance, and healthcare, where they are used to power personalized recommendations, fraud detection, and disease diagnosis.

Knowledge Graphs have the potential to play a significant role in the field of XAI by providing a high-level, conceptual representation of the data and relationships between entities. By using Knowledge Graphs to represent the inputs to ML models and their predictions, it becomes possible to understand the reasoning behind a particular prediction, as well as to present this information in a form that is easy for humans to understand

2 Literature Review

The main purpose of this section is to review the existing literature on the use of knowledge graphs as an eXplainable AI method to explain ML models and their predictions, and to discuss the key findings and contributions of each study.

Our emphasis is on applications that are based on Machine Learning. We have categorized the works based on their generic application domain, ranging from image classification to item recommendation, natural language processing, predictive tasks, and healthcare.

2.1 Computer Vision

Numerous works have shown early on how Knowledge Graphs can be used to improve performance and provide knowledge-based explanations on Computer Vision tasks, such as image recognition or image retrieval.

One such example is given in [1], where commonsense knowledge bases from ConceptNet are used to improve the performance on sentence-based image retrieval tasks. Consider an image where you have a cook, a *Chef* who's getting ready to stir up some stir fry in the pan. Traditional image retrieval techniques require visual anchors on the entity *Chef* to recognize it. However, since on the image there are no visual detectors related to the profession itself, a data-driven approach would be needed to expose the model to recurring visual patterns across a sufficient number of samples until it is able to recognize the presence of a *Chef* in the image. The approach provided by the authors shows how the relations between *Chef* and words like *Kitchen*, *Dish* and *Person* that can be found in ConceptNet's ontology provide visual detectors (related to the concept Chef) that greatly facilitate the retrieval of the corresponding image. Such an example on the MS-Coco dataset can be found on Figure 1. Furthermore, we can see how this technique can be used for downstream explanation-oriented tasks, since associating an image to a representational KG can provide justification as to why a certain image was retrieved based on the query sentence.

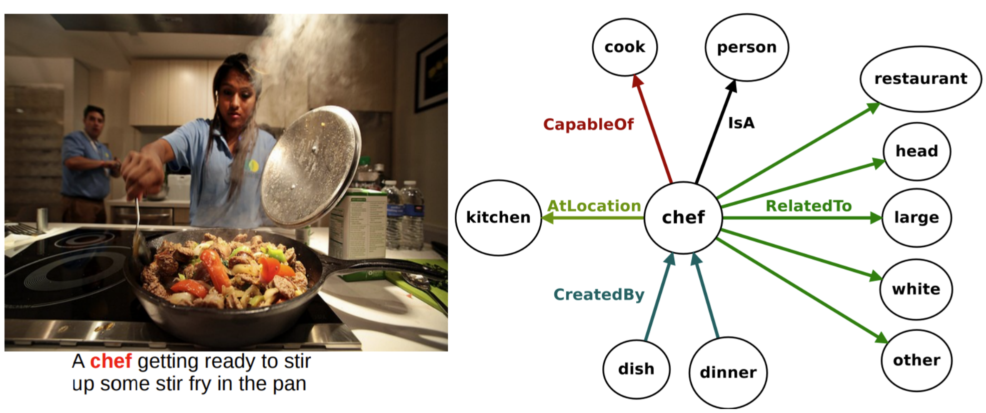

Figure 1: **Left**. An image and one of its associated sentences from the MS COCO dataset. Among its words, the sentence features the word Chef, for which there is not a visual detector available. **Right**. Part of the hypergraph at distance 1 related to word Chef in ConceptNet. In the list of nodes related to the concept Chef, there are several informative concepts for which we have visual detectors available.

Similar techniques can be seen for image recognition tasks. The authors in [2] use a combination of a general Deep Neural Network (DNN) structure and WordNet to classify scenes. They match the object categories from WordNet to those in the ADE20K dataset and then train an object recognition component using WordNet's hierarchy. This recognition component is then input into a linear regression model which can provide human-readable explanations automatically. [3] presents the use of non-propositional rules from a knowledge graph for multiclass prediction in the context of object detection. This is achieved by integrating a Convolutional Neural Network (CNN) architecture with an Inductive Logic Programming (ILP) framework. The ILP framework enables the learning of OWL class expressions based on ontological rules as background knowledge.

2.2 Recommender Systems

Recently, the use of knowledge graphs to improve the transparency of recommender system results has gained popularity with the aim of improving user satisfaction, trust, and loyalty. The majority of these approaches are content-based, meaning they provide explanations for recommendations using entities from a specified knowledge graph, often presented in the form of images or natural language sentences.

In reference [4], the authors suggest replacing the hidden layers and connections of an autoencoder neural network with the structure of DBpedia, conforming to the Semantics-Aware Autoencoders [5]. To define movies, they utilize a limited set of predicates (dct:subject, dct:starring, dct:director, dct:writer) and generate human-readable explanations by relying on the weights related to the features in the user's profile. Explanations are provided in three forms, namely, popularity-based ("We recommend X and Y as they are highly popular among individuals with similar movie preferences as yours"), pointwise customization ("We believe you would enjoy watching movies that are related to X and Y"), and pairwise customization ("We think you would prefer X over Y as you might have a preference for x_i"). An example is given in Figure 2.

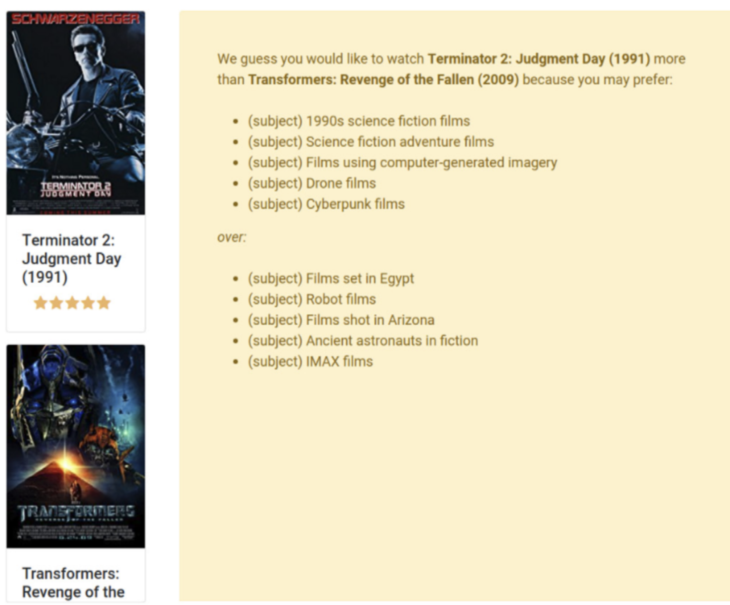

Figure 2: Pairwise explanations of recommended movies using *dc:subject* relationships.

In the same vein, [6] concentrates on recommending Amazon products by merging a collaborative filtering technique with a knowledge graph. The model incorporates an ad-hoc graph of entities and user behaviors along with a minimal set of properties (e.g. produced by, category, also viewed) that are automatically constructed. Personalized recommendations and their natural language explanations are then generated using a soft-matching algorithm applied to the knowledge graph. The result is a much more detailed justification into why a certain product has been recommended to a client, a major advantage for recommendation systems who are notoriously known for being opaque (black-box). This leap in transparency can be viewed in Figure 3, in which an explanation can be seen through the interpretation of the graph: *Bob* has previously mentioned the word "IOS" and has also bought another product that was produced by Apple, which is why he was proposed an iPad.

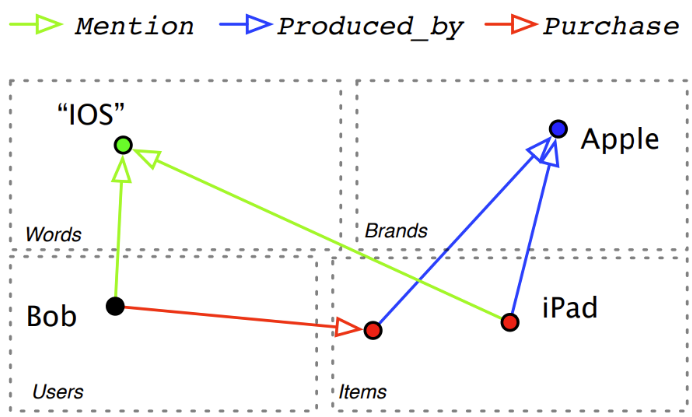

Figure 3: Explanation paths between a user *Bob* and the recommended item *iPad*.

Such explanation techniques can greatly aid in gaining more trust from the customer side, and data-driven techniques can be seen as more transparent. This could provide an answer to the challenge of the perception of recommendation systems by the public opinion, since a lot of people still feel as if their privacy is breached when products are recommended because of "*user collected data*"; a thorough justification and more transparent approach to the customer could bridge this psychological gap.

The literature on explainable recommender systems discussed in this section shares similarities with the systems discussed in the previous section, such as the integration of knowledge graphs and the provision of explanations in written or visual form. The main differences lie mainly in the use of large-scale, open-domain knowledge graphs (primarily DBpedia) to gather additional information about the input data to provide user-friendly explanations, and the absence of the use of axioms. This suggests a challenge in scaling up inference techniques [7]. However, the use of open-domain knowledge graphs also poses problems such as information overload, where the high outdegree of nodes restricts the selection of relevant information, often presented as multi-edge paths extracted from the graphs (as seen in Fig. 3). This not only impedes a more comprehensive knowledge discovery process, but also often necessitates careful pruning to provide explanations that are more trustworthy for end-users.

2.3 NLP

A significant area of research can be observed in the realm of natural language applications, including Knowledge-Based Question-Answering (KB-QA), Machine Reading Comprehension, and Conversational AI. These fields often utilize knowledge graphs as a source of common-sense knowledge to answer questions presented in various formats, including images, speech, and text.

Taking a closer look into applications in KB-QA, [8] utilizes ConceptNet as background information in order to answer questions in a specific field, such as scientific inquiries. This is done through the combination of query reformulation, structured background knowledge, and textual entailment to provide a clearer answer. Similarly, [9] utilizes ConceptNet to provide commonsense links between concepts for questions answered by QA models.

Another paper, [10], uses open-domain knowledge graphs to answer single-fact questions. This is achieved by converting the questions into structured SPARQL queries, which are then processed by a Gated Recurrent Unit recurrent neural network. This method was trained on the SimpleQuestions dataset, which consisted of around 110,000 English questions paired with subject-relation-object triples from Freebase. The work of [11] extends this idea by incorporating multiple interconnected knowledge graphs to improve open-domain question-answering. The researchers extract candidate triple patterns from natural language questions, then align them with a joint inference model based on integer linear programming. This is used to retrieve answers and additional information from the knowledge graphs.

Additionally, [12] explores the use of knowledge-based explanations for visual question-answering (VQA). This work combines DBpedia, WebChild, and ConceptNet to extract triple patterns that support answers to visual questions. A combined RNN-LSTM model is used to extract relevant information from the knowledge graphs for the input images. Unlike current approaches in VQA, this method allows for explicit reasoning, where answers are seen as indicative explanations for the response. As can be seen on Figure 4, additional supporting facts are given in order to explain the answer given to the question. Similarly to applications mentioned in previous sections, information extracted from KGs are used to elicit the reasoning and further increase transparency.

Question: What can the red object on the ground be used for ?
Answer: Firefighting
Support Fact: Fire hydrant can be used for fighting fires.

KB	Relationship	#Facts	Examples
DBpedia	Category	35152	(Wii,Category,VideoGameConsole)
ConceptNet	RelatedTo	79789	(Horse,RelatedTo,Zebra), (Wine,RelatedTo,Goblet), (Surfing,RelatedTo,Ocean)
	AtLocation	13683	(Bikini,AtLocation,Beach), (Tap,AtLocation,Bathroom)
	IsA	6011	(Broccoli,IsA,GreenVegetable)
	CapableOf	5837	(Monitor,CapableOf,DisplayImages)
	UsedFor	5363	(Lighthouse,UsedFor,SignalingDanger)
	Desires	3358	(Dog,Desires,PlayFrisbee), (Bee,Desires,Flower)
	HasProperty	2813	(Wedding,HasProperty,Romantic)
	HasA	1665	(Giraffe,HasA,LongTongue), (Cat,HasA,Claw)
	PartOf	762	(RAM,PartOf,Computer), (Tail,PartOf,Zebra)
	ReceivesAction	344	(Books,ReceivesAction,bought at a bookshop)
	CreatedBy	96	(Bread,CreatedBy,Flour), (Cheese,CreatedBy,Milk)
WebChild	Smaller, Better, Slower, Bigger, Taller, ...	38576	(Motorcycle,Smaller,Car), (Apple,Better,VitaminPill), (Train,Slower,Plane), (Watermelon,Bigger,Orange), (Giraffe,Taller,Rhino), (Skating,Faster,Walking)

Figure 4: Supporting facts extracted from ConceptNet and WebChild used to explain anwsers.

2.4 Predictive and Forecasting tasks

In this section, we'll look at how knowledge graphs can be used to help understand and explain predictions made in areas such as loan applications, market analysis, and traffic dynamics. The idea behind these systems is that by connecting raw input data to nodes in the graph, more information about the data can be obtained by exploring the graph.

In [13], the authors show how ontologies help the understandability of interpretable machine learning models, such as decision trees. In particular, they build on Trepan [14], an algorithm that explains artificial neural networks by means of decision trees, and extend it to include ontologies modeling domain knowledge in the process of generating explanations. An example is given for the context of loan prediction in Figure 7.

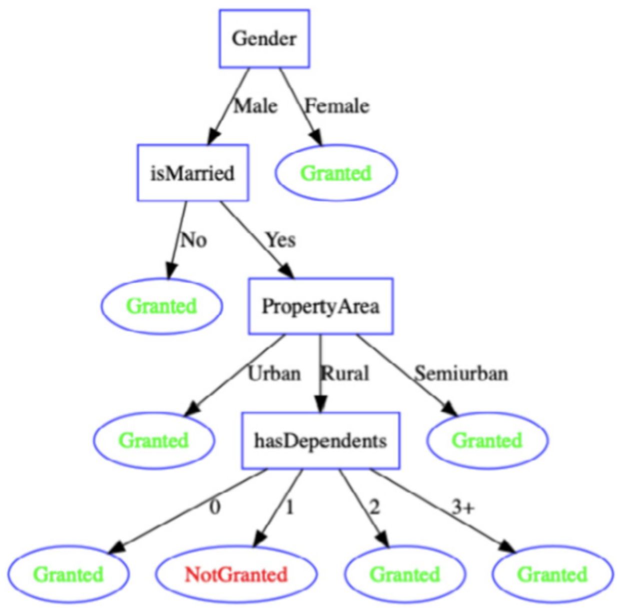

Figure 5: Decision tree using domain ontology concepts to explain the conditions to grant or refuse loans.

In the study done by [15], the use of knowledge graph embeddings through CrossE, the novel knowledge graph embeddings developed by the authors, is explored to find explanations for predicted links in a knowledge graph completion task. The explanations are seen as closed paths between the head and tail entities of the predicted link, and the embeddings allow for the identification of the most accurate paths in terms of recall and average support. Another study, [16], proposes a knowledge graph-based transfer learning method to explain why certain flights may be delayed. This approach involves learning predictions from a dataset and a local OWL ontology, then using TBox assertions from DBpedia to explain positive and negative transfers from one domain to another. Lastly, [17] suggests using RDFS ontologies to enhance the input data for binary classifiers by converting the data into concepts that can be used to create explanations that are easy for people to understand. The concepts are taken from DBpedia and the Microsoft Concept Graph and then mapped to a domain ontology.

2.5 Healthcare

Knowledge graphs are applied in different ways in healthcare to increase the explainability of AI models. They annotate, organize, and present different types of information meaningfully and add semantic labels to healthcare datasets.

Some examples of KG applications for Medical XAI are:

- **Entity/Relation Extraction**: Clinical notes usually contain narratives of patients' interactions, which are often presented as unstructured data and free text in healthcare. This information is transformed into a structured format, such as a named entity or common vocabulary, through the use of knowledge graphs. Named entity recognition methods are utilized to represent clinical notes and map them to vocabularies using named entity normalization techniques. Additionally, knowledge graphs are utilized in relation extraction, where the semantic relation between two entities is typically extracted [18]. For example, a disease knowledge graph can be used to extract the relationship between a disease and other concepts, such as diagnosis or treatment, through the application of different relation extraction algorithms.
- **Inference and Reasoning**: Knowledge graphs typically utilize deduction reasoning to infer new facts and knowledge. Reasoning over a knowledge graph is an evidence-based approach that is more acceptable and interpretable for clinicians. For instance, EHR (electronic health record) data can be transformed into a semantic net model (patient-centered) under a knowledge graph, allowing for the creation of an EHR data trajectory and reasoning using semantic rules. This type of system design allows for reasoning to identify critical clinical discoveries within EHR data and presents the clinical significance to clinicians, providing a better understanding of neglected information [19]. An example of an EHR KG can be seen in Figure 6.
- **Explanations and Visualizations**: XAI models provide explanations for physicians and healthcare professionals, making the outputs understandable and transparent. Knowledge graphs also help provide further insights into the reasons behind model predictions and can represent the results in graphs. Human-in-the-loop techniques can be utilized to validate the results or refine the knowledge graph, leading to high accuracy and better explainability.

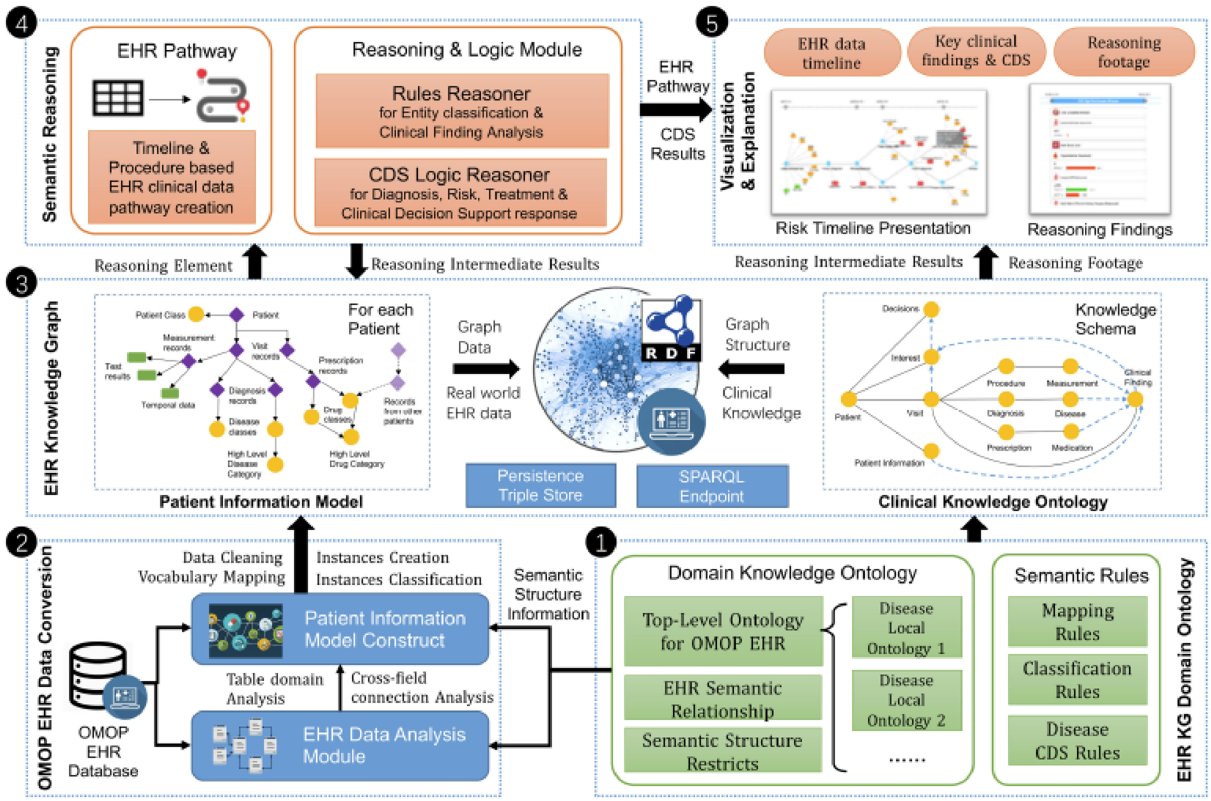

Figure 6: Knowledge Graph System Architecture. (1) The EHR KG domain ontology module defines the semantic structure of the system and the reasoning logics. (2) Analysis of EHR data and transformation into an RDF-type patient information graph. (3) Storage of medical knowledge and EHR data in the knowledge graph. (4) Performing semantic reasoning and creating CDS results. (5) Visualization of the reasoning footage and EHR data for result interpretation and explanation.

3 Case Study: Explainable Recommender Systems

Recommender systems have become increasingly precise in recent years, leveraging sophisticated algorithms and large amounts of data to make personalized recommendations to users. [4] developed a set of metrics to evaluate recommender systems based on different criteria: *satisfaction*, *effectiveness*, *trust*, *persuasiveness* and *transparency*. The authors show that some systems perform very well in terms of effectiveness but fail to appear as trustworthy or transparent to users. Indeed, despite these models' success in improving the accuracy of recommendations, there is a growing demand for transparency and *explainability* in recommender systems. Knowledge-Base Embeddings (KBE) allow to embed heterogeneous entities (i.e. users, items, words, etc.) in such a way that models are able to learn user and items representations without compromising the external knowledge needed to generate the explanation for the recommended items [6].

3.1 Collaborative Filtering (CF) over Knowledge Graphs

Collaborative filtering is a popular approach for building recommender systems. The idea behind it is to make recommendations based on the past behavior and preferences of users. The two main types of collaborative filtering are *user-user collaborative filtering* and *item-item collaborative filtering*. The first one outputs recommendations to a given user by suggesting items that similar users have liked, while the second one recommends certain items that are similar to the items a given user has liked in the past.

However, CF-based models do not take advantage of the ability of knowledge graphs to represent more complex relationships between entities. Burke [20] and Trewin [21] introduced a new kind of recommender system that leverages the relationships between entities in the graph: a hybrid system combining the recommendation capabilities of CF-based models with the explainability power of knowledge-based models. Thus, applying CF over knowledge graphs allows for a more nuanced and sophisticated representation of the data, which not only improves the recommendation performance, but also preserves the external knowledge structure to explain the reasoning behind the recommendation.

3.2 Steps to build a KBE-based CF model

While many have tried to leverage the knowledge-base embeddings for recommendation [24,25], few have used this framework to generate explainable recommendations [6]. In KBE, entities are represented as latent vectors and relations as a linear transformation from one entity to another. This connection between entities is computed in a soft manner using their embeddings, as discussed in the following.

3.2.1 Problem Formulation

The goal of a KBE-based CF model here is twofold: it has to *recommend* items to users (for each user, it has to find a set of the items most likely to be purchased) but also be able to *explain* the reasons behind its recommendation (by forming a natural language sentence explain each recommendation). First of all, the knowledge-base is represented as a set of triplets:

$$S = \{(e_h, e_t, r)\}$$

where e_h and e_t are respectively a head entity and a tail entity, and r is the relation from e_h to e_t. The different types of entities are *user*, *item*, *brand*, etc., and the relationships are *Purchase*, *Mention*, *Produced_ by*, *Also_ bought*, among others.

The problem now becomes:

1. For each user u, find the products i that are most likely to be linked with u by the *Purchased* relationship.
2. For each recommendation (u, i), provide an explanation based on the relationships and entities related to the pair in the form of a easy-to-understand natural language sentence.

3.2.2 Construction process of the KG

Each entity is represented in a low-dimensional space and relations between entities are treated as translation functions that convert one entity to another. Entities e_h and e_t are represented as latent vectors, based on a previous approach [22]. Moreover, their relationship r is being modeled as a linear projection from e_h to e_t parameterized by $r \in \mathbb{R}^d$, namely:

$$e_t = \mathbf{trans}(e_h, r) = e_h + r \tag{1}$$

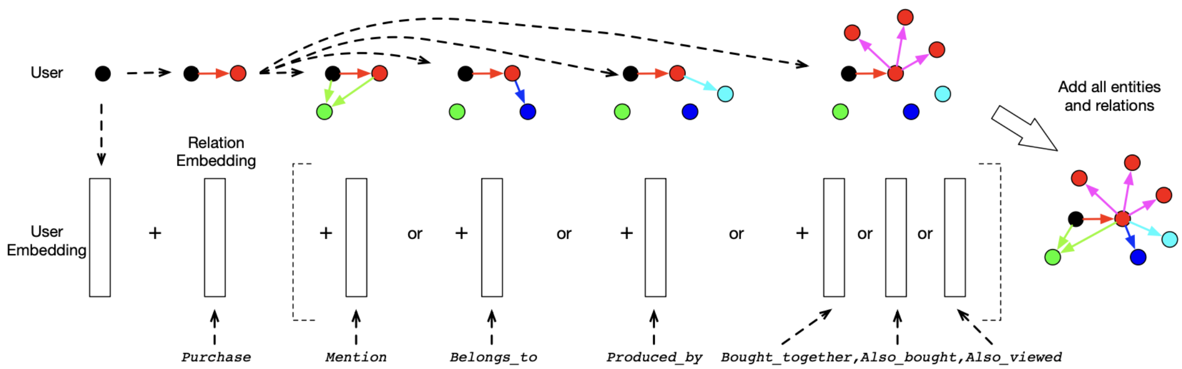

Figure 7: The construction process of knowledge graph. Each entity is represented with a latent vector, and each relation is modeled as a linear translation from one entity to another entity parameterized by the relation embedding. [6]

Mikolov et al. introduced the embedding-based generative framework, that is being used in this model in order to learn the translation model $\mathbf{trans}(e_h, r)$ and which has been widely used for word embedding [23, 24], recommendation [25, 26], and information retrieval [27, 28]. To be specific, when given a relation triplet $(e_h, e_t, r) \in S$, learning the translation model is done by optimizing the following probability [6]:

$$P(e_t|\mathbf{trans}(e_h, r)) = \frac{\exp(e_t \times \mathbf{trans}(e_h, r))}{\sum_{e'_t \in E_t} \exp(e'_t \times \mathbf{trans}(e_h, r))} \tag{2}$$

where E_t is the set of all possible entities that share the same type with e_t.

By doing this, Equation (1) is transformed into an optimization problem that can be solved using iterative optimization algorithms like gradient descent.

For model optimization, the representations of entities and relations are learned by maximizing the sum of the log-likelihood - a computationally-efficient reformulation of the likelihood - of all observed relation triplets.

It is given in its approximate form by [6]:

$$\mathcal{L}(S) = \sum_{(e_h, e_t, r) \in S} \log \sigma(e_t \times \mathbf{trans}(e_h, r)) + k \times \mathbb{E}_{e'_t \sim P_t}(\log \sigma(-e'_t \mathbf{trans}(e_h, r)) \tag{3}$$

3.2.3 Recommendation Explanation with KBE

In order to create explanations for given recommendations, we need to define *explanation paths* as well as a method to generate natural language explanations with said paths. An explanation path is a sequence from the user u to the item i in the latent space. There is an explanation path between entity e_u and entity e_i if there exists an entity e_x that can be inferred by both e_u and e_i using their respective sets of relations $R_\alpha = \{r_\alpha\}$ and $R_\beta = \{r_\beta\}$ such that [6]:

$$e_u + \sum_\alpha r_\alpha = e_x = e_i + \sum_\beta r_\beta \tag{4}$$

The simplest illustration of such a path is given in 3, where Bob (e_u) and iPad (e_i) are both connected to the word "IOS" by the *Mention* relation.

Since the latent representations of entities and relationships are learned during the optimization of the model, we can overcome the sparsity of the knowledge graphs by extending the softmax function to compute $P(e_x|\mathbf{trans}(e_u, R_\alpha))$ (i.e. performing *entity soft matching*) and are thus able to obtain the probability of explanation paths, given by [6]:

$$P(e_x|\mathbf{trans}(e_u, R_\alpha, e_i, R_\beta)) = P(e_x|\mathbf{trans}(e_u, R_\alpha)) \times P(e_x|\mathbf{trans}(e_i, R_\beta)) \tag{5}$$

We can now rank these paths by decreasing values of $P(e_x|\mathbf{trans}(e_u, R_\alpha, e_i, R_\beta))$, which in turn allows us to provide the explanation for the recommendation in natural language. To do so, we select the best path and use predefined templates to write a sentence describing it. In the case of 3, assuming that the best path was the one linking both *Bob* and *iPad* to "IOS", the explanation could be something like: "***Bob*** *may be interested in* ***iPad*** *because he (**Bob**) often* ***mentions 'IOS'*** *in his reviews, and* ***'IOS'*** *is often* ***mentioned*** *in the reviews of* ***iPad***" [6].

4 Conclusion

In this paper, we made a review of the different works that use Knowledge Graphs in order to make AI more explainable. We went across the domains of Computer Vision, Recommender Systems, NLP, Predictive and Forecasting tasks and Healthcare.

In the second part of this paper, we studied the specific case of explainable recommendation systems. The work we analysed uses a method to learn over heterogeneous Knowledge-Based Embeddings (KBE) for personalized and explainable recommendations. To achieve this, an user-item knowledge graph which combines both user behaviors and knowledge about the items is constructed. Then, the heterogeneous relations are used collectively to learn the KBE and generate personalized recommendations thanks to the user and item embeddings. To provide explanations about the recommendations, a soft matching algorithm is used to identify explanation paths between a user and the recommended items in the latent KBE space.

Bibliography

[1] Rodrigo Toro Icarte, Jorge A Baier, Cristian Ruz, and Alvaro Soto. How a general-purpose commonsense ontology can improve performance of learning-based image retrieval. *arXiv preprint arXiv:1705.08844*, 2017.

[2] Zachary A Daniels, Logan D Frank, Christopher J Menart, Michael Raymer, and Pascal Hitzler. A framework for explainable deep neural models using external knowledge graphs. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II*, volume 11413, pages 480–499. SPIE, 2020.

[3] Md Kamruzzaman Sarker, Ning Xie, Derek Doran, Michael Raymer, and Pascal Hitzler. Explaining trained neural networks with semantic web technologies: First steps. *arXiv preprint arXiv:1710.04324*, 2017.

[4] Vito Bellini, Angelo Schiavone, Tommaso Di Noia, Azzurra Ragone, and Eugenio Di Sciascio. Knowledge-aware autoencoders for explainable recommender systems. In *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, pages 24–31, 2018.

[5] Vito Bellini, Tommaso Di Noia, Eugenio Di Sciascio, and Angelo Schiavone. Semantics-aware autoencoder. *IEEE Access*, 7:166122–166137, 2019.

[6] Qingyao Ai, Vahid Azizi, Xu Chen, and Yongfeng Zhang. Learning heterogeneous knowledge base embeddings for explainable recommendation. *Algorithms*, 11(9):137, 2018.

[7] Heiko Paulheim and Aldo Gangemi. Serving dbpedia with dolce–more than just adding a cherry on top. In *The Semantic Web-ISWC 2015: 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I 14*, pages 180–196. Springer, 2015.

[8] Ryan Musa, Xiaoyan Wang, Achille Fokoue, Nicholas Mattei, Maria Chang, Pavan Kapanipathi, Bassem Makni, Kartik Talamadupula, and Michael Witbrock. Answering science exam questions using query rewriting with background knowledge. *arXiv preprint arXiv:1809.05726*, 2018.

[9] Wanjun Zhong, Duyu Tang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. Improving question answering by commonsense-based pre-training. In *Natural Language Processing and Chinese Computing:*

8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part I 8, pages 16–28. Springer, 2019.

[10] Zihang Dai, Lei Li, and Wei Xu. Cfo: Conditional focused neural question answering with large-scale knowledge bases. *arXiv preprint arXiv:1606.01994*, 2016.

[11] Yuanzhe Zhang, Shizhu He, Kang Liu, and Jun Zhao. A joint model for question answering over multiple knowledge bases. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[12] Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton Van Den Hengel. Fvqa: Fact-based visual question answering. *IEEE transactions on pattern analysis and machine intelligence*, 40(10):2413–2427, 2017.

[13] Roberto Confalonieri, F Moscoso del Prado, Sebastia Agramunt, Daniel Malagarriga, Daniele Faggion, Tillman Weyde, and Tarek R Besold. An ontology-based approach to explaining artificial neural networks. *arXiv preprint arXiv:1906.08362*, 2019.

[14] Roberto Confalonieri, Tillman Weyde, Tarek R Besold, and Fermín Moscoso del Prado Martín. Trepan reloaded: A knowledge-driven approach to explaining artificial neural networks. 2020.

[15] Wen Zhang, Bibek Paudel, Wei Zhang, Abraham Bernstein, and Huajun Chen. Interaction embeddings for prediction and explanation in knowledge graphs. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 96–104, 2019.

[16] Jiaoyan Chen, Freddy Lécué, Jeff Z Pan, Ian Horrocks, and Huajun Chen. Knowledge-based transfer learning explanation. In *Sixteenth International Conference on Principles of Knowledge Representation and Reasoning*, 2018.

[17] Freddy Lécué and Jiewen Wu. Semantic explanations of predictions. *arXiv preprint arXiv:1805.10587*, 2018.

[18] Hasham Ul Haq, Veysel Kocaman, and David Talby. Deeper clinical document understanding using relation extraction. *arXiv preprint arXiv:2112.13259*, 2021.

[19] Yong Shang, Yu Tian, Min Zhou, Tianshu Zhou, Kewei Lyu, Zhixiao Wang, Ran Xin, Tingbo Liang, Shiqiang Zhu, and Jingsong Li. Ehr-oriented knowledge graph system: toward efficient utilization of non-used information buried in routine clinical practice. *IEEE Journal of Biomedical and Health Informatics*, 25(7):2463–2475, 2021.

[20] Robin Burke. Integrating knowledge-based and collaborative-filtering recommender systems. In *Proceedings of the Workshop on AI and Electronic Commerce*, pages 69–72, 1999.

[21] Shari Trewin. Knowledge-based recommender systems. *Encyclopedia of library and information science*, 69(Supplement 32):180, 2000.

[22] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[24] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

[25] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1449–1458, 2017.

[26] Lei Zheng, Vahid Noroozi, and Philip S Yu. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 425–434, 2017.

[27] Qingyao Ai, Liu Yang, Jiafeng Guo, and W Bruce Croft. Analysis of the paragraph vector model for information retrieval. In *Proceedings of the 2016 ACM international conference on the theory of information retrieval*, pages 133–142, 2016.

[28] Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W Bruce Croft. Learning a hierarchical embedding model for personalized product search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 645–654, 2017.