Ministère de l'enseignement Supérieur et de la recherche Scientifique
وزارة التعليم العالي والبحث العلمي

Université Badji Mokhtar – Annaba — **جامعة باجي مختار ـعنابـــــــــة**

Faculté de Technologie — **كلية علـــــــوم الهندســـــــة**

Département d'Informatique — **قســـــــم الإعلام الآلي**

Thèse

Présentée pour obtenir le diplôme de

Doctorat 3ème Cycle

Spécialité : Sciences et Technologies de l'Information et de la Communication

Filière : Informatique

Par :
BOUGHAREB Rima

Thème :

Étude et Analyse des Méthodes d'Apprentissage pour les Données Sémantiques à Grande Échelle

Thèse soutenue le ….. date de soutenance ……. devant le jury composé de :

N°	Nom et prénom	Grade	Etablissement	Qualité
01	Souici-Meslati Labiba	Prof.	Université Badji Mokhtar -Annaba	Président
02	Seridi Hassina	Prof.	Université Badji Mokhtar -Annaba	Rapporteur
03	Beldjoudi Samia	MCA	Université Badji Mokhtar -Annaba	Co-rapporteur
04	Farah Nadir	Prof.	Université Badji Mokhtar -Annaba	Examinateur
05	Halimi Khaled	MCA	Université 8 mai 1945 - Guelma	Examinateur
06	Khebizi Ali	MCA	Université 8 mai 1945 - Guelma	Examinateur

This thesis is lovingly dedicated to my family…

Acknowledgements

I would like to express my deepest gratitude to my advisor, Professor Hassina Seridi, for her unwavering guidance, patience, and encouragement throughout the course of this Ph.D. thesis. Her expertise and insights have been invaluable in shaping the direction of my research and pushing me to achieve my best.

I am also grateful to my co-supervisor, Dr. Samia Beldjoudi, for her assistance and valuable contributions. Her valuable input and collaboration and feedback have been instrumental in the development and refinement of this research.

I would like to extend my heartfelt appreciation to Dr. Djalila Boughareb for her assistance and support.

Finally, I would like to express my gratitude to all the professors, researchers, and colleagues who have provided guidance, and engaged in insightful discussions, which have greatly contributed to the development of this thesis.

Thank you all for your invaluable contributions and for being an integral part of this challenging and rewarding endeavor.

ABSTRACT

The expansion of the World Wide Web (WWW) has significantly changed how people produce and consume information. However, it is evolving beyond being a simple web of documents that only people can understand. With the advent of the Semantic Web, the web's capabilities have been further enhanced by moving from a web of interconnected web pages primarily intended for human consumption to a more sophisticated web of Linked Data that is both machine-readable and human-understandable. This evolution has unlocked new possibilities for knowledge integration, discovery, and automation particularly since Tim Berners-Lee, the mind behind the Web, introduced the Linked Open Data (LOD) Principles. The primary objective of the LOD initiative is to promote the open sharing and interlinking of diverse datasets. LOD allows for the creation of a vast network of interconnected information, facilitating seamless data integration and interoperability across different domains. The foundation of this data landscape rests upon Knowledge Graphs (KGs), serving as the building blocks of the LOD cloud and heralding a new era in Knowledge Representation. Recommender Systems (RSs), being data-driven tools, can greatly benefit from the use of KGs, as they provide access to more detailed information about items. In fact, KGs offer a wide range of interconnected and semantically-rich data, providing valuable insights into user preferences and item features. This, in turn, can lead to higher user satisfaction. The crucial point to incorporate KGs to RSs is how to design an effective representation of the semantic information embedded within high-order connections that requires careful consideration and planning to ensure that the model can accurately capture, fully exploit, and utilize the relevant information in a meaningful way. Furthermore, the integration of KGs brings forth another challenge: handling the complex patterns present within the graph. Graph embedding emerges as a promising solution to address this challenge. Graph Embedding techniques are class of Machine Learning algorithms that aim to learn low-dimensional representations of the graph, which can then be used to solve a variety of downstream tasks. This Ph.D. thesis resides at the crossroads of two research domains: Recommender Systems and Semantics. We demonstrate how the latest advancements in Machine Learning can be leveraged to learn features from Knowledge Graphs for recommendation purposes. In our first contribution, we propose a Knowledge Representation that aligns with both Knowledge Graph and the user-item bipartite graph. This representation involves converting properties (edges) into entities (nodes), allowing us to preserve the semantics conveyed by the predicates during the learning phase. To enable recommendations, we apply Node2vec, a Graph Representation Learning algorithm, to jointly learn embeddings for entities and semantic properties. By incorporating properties into the embedding process, we gain a deeper understanding of user preferences and tackle the diversity problem. We evaluate two recommendation methods, one inspired by the TransE method and the other based on measuring user-item relatedness using cosine similarity. The results demonstrate improved performance in terms of precision and diversity. Our second contribution introduces Explain-KGCN, an Explainable KG-aware Recommender System. First, we design a weighted Knowledge Representation based on a novel relational weighting metric specifically designed for KGs. This metric incorporates prior information about the importance of nodes in the KG by assigning personalized weights to each node. We also leverage richer information about user preferences by considering both positive and negative explicit user feedback. Additionally, we develop a property-specific algorithm based on a Graph Convolutional Network (GCN), a robust Deep Learning Technique for Graph Embedding. The core component of our algorithm is a relation-specific GCN aggregator function that extracts low-dimensional representations for each node in the Knowledge Graph by gathering data from its relation-specific

neighbors. This approach enables us to model the semantics of different relations more explicitly, leading to better quality embeddings, more accurate predictions, and precise explanations. Building upon the previous contributions, our third contribution automates the weighting process using Graph Attention Networks (GATs), a commonly-used architecture in Deep Graph Neural Networks for recommendation. We explore the potential of KGs for interpretable recommendation using GATs, aiming to fully exploit the semantic information and preserve the inherent knowledge encapsulated in relations. By incorporating additional knowledge from the LOD cloud and utilizing GATs, we generate vector representations for each node in the graph. The proposed approach improves predictive performance in terms of precision, recall, and diversity, fully harnessing the rich structured information provided by KGs to offer explanations for recommendations across three different domains.

Keywords. *Linked Data, Graph Embedding, Machine Learning, Knowledge Graph, Knowledge Representation, Semantic Web, Recommender Systems, Explainability, Diversity.*

RÉSUMÉ

L'expansion du World Wide Web (WWW) a considérablement modifié la façon dont les gens produisent et consomment de l'information. Avec l'avènement du Web Sémantique, les capacités du web ont été renforcées en passant d'un Web d'une génération basée sur les documents exclusivement compréhensible par les humains, vers une génération basée sur les données où l'information est devenue à la fois lisible et compréhensible par les humains et par les machines. Cette évolution a ouvert de nouvelles possibilités d'intégration des connaissances, en particulier depuis que Tim Berners-Lee, le concepteur du Web, a introduit les principes des Données Ouvertes Liées (Open Linked Data : LOD). L'objectif principal de l'initiative LOD est de promouvoir le partage et l'interconnexion des données. Ce qui a permis la création d'un vaste réseau d'informations interconnectées qui facilitent l'intégration et l'interopérabilité des données. Les Graphes de Connaissances (Knowledge Graphs : KGs) constituent les éléments fondateurs du nuage LOD, une nouvelle ère dans la représentation des connaissances est annoncée. Les Systèmes de Recommandation (SRs), en tant que outils basés sur les données, peuvent bénéficier considérablement des KGs. En fait, les KGs offrent une large gamme de données interconnectées qui peuvent aider à une modélisation sémantiquement riche des préférences des utilisateurs. Le point crucial pour intégrer les KGs aux SRs réside dans la manière de concevoir une représentation efficace des données sémantiques, ce qui nécessite une réflexion et une planification attentives pour garantir la capture avec précision, l'exploitation et l'utilisation de manière significative ces informations pertinentes. De plus, l'intégration des KGs présente un défi supplémentaire : la gestion des structures complexes résultant de cette intégration. Les techniques d'incorporation de graphes (Graph Embedding : GE) émergent comme une solution prometteuse pour relever ce défi. Les GEs sont une classe d'algorithmes d'Apprentissage Automatique qui visent à apprendre des représentations vectorielles qui peuvent être utilisées pour résoudre de différents problèmes. Cette thèse se positionne à l'intersection de deux domaines de recherche : les SRs et la représentation sémantique. Nous démontrons comment les dernières tendances des techniques d'Apprentissage Automatique peuvent être exploitées pour apprendre des caractéristiques à partir des graphes de connaissances pour leur utilisation efficiente dans les systèmes de recommandation. Dans notre première contribution, nous avons proposé une représentation des connaissances qui s'harmonise à la fois avec les graphes de connaissances et le graphe utilisateur-item. Cette représentation consiste à convertir les propriétés (arêtes) en entités (nœuds), ce qui nous a permis de préserver la sémantique portée sur les prédicats pendant la phase d'apprentissage. Nous avons utilisé Node2vec, un algorithme d'apprentissage de représentation de graphe, pour générer des représentations pour les entités et les propriétés sémantiques. Nous adoptons deux stratégies de recommandation, l'une inspirée de la méthode TransE et l'autre basée sur la mesure cosinus. Les résultats démontrent une amélioration des performances en termes de précision et de diversité. Notre deuxième contribution présente Explain-KGCN, un SR explicatif basé sur les Graphes de connaissances et les Graphes Neuronales Convolutionnels (Graph Convolutional Networks : GCN). Nous avons proposé en premier une représentation pondérée des connaissances basée sur une nouvelle métrique de pondération relationnelle spécialement conçue pour pondérer les KGs. Cette métrique incorpore des poids personnalisés pour chaque nœud en fonction d'informations préalables sur son importance. Nous exploitons également des informations plus riches sur les préférences des utilisateurs en tenant compte des évaluations des utilisateurs explicites positives et négatives. En s'appuyant sur les contributions précédentes, notre troisième contribution automatise le processus de pondération en utilisant des réseaux d'attention de graphe (Graph Attention Networks : GATs). Nous explorons le potentiel des KGs pour fournir des explications pour les recommandations dans le but d'exploiter pleinement les informations sémantiques et de préserver les connaissances inhérentes encapsulées dans les propriétés. L'approche proposée améliore les performances prédictives en termes de précision, de rappel et de diversité, exploitant pleinement les informations structurées riches fournies par les KGs grâce aux GATs pour offrir des explications pour les recommandations dans trois domaines différents.

Mots-clés. *Données liées, Incorporation de graphe, Apprentissage automatique, Graphe de connaissances, Représentation des connaissances, Web sémantique, Systèmes de recommandation, Explicabilité, Diversité.*

ملخـــــص

توسّع الويب قد غير بشكل كبير كيفية إنتاج المعلومات واستهلاكها. مع ظهور الويب الدلالي، تم تعزيز قدرات الويب بشكل أكبر من خلال الانتقال من شبكة من صفحات الويب المتصلة تم تصميمها بشكل أساسي للاستهلاك البشري إلى شبكة متطورة من البيانات المرتبطة قابلة للقراءة والفهم بواسطة الآلة والبشر. هذه التطورات فتحت إمكانيات جديدة لدمج المعرفة واكتشافها، خاصةً منذ أن قام تيم بيرنرز-لي، العقل المبدع وراء الويب، بتقديم مبادئ البيانات المفتوحة المتصلة .(LOD) الهدف الأساسي لمبادرة LOD هو تعزيز مشاركة وربط مجموعات البيانات المتنوعة بشكل مفتوح، حيث يسمح LOD بإنشاء شبكة واسعة من المعلومات المترابطة، مما يسهل دمج البيانات بسلاسة والتوافق عبر مجالات مختلفة. تقوم أساسيات هذا النظام البياني على شبكات المعرفة (Knowledge Graphs) حيث تعتبر هذه المكونات أساساً لسحابة LOD وتعلن عن عصر جديد في تمثيل المعرفة. يمكن لأنظمة التوصية (RSs) كأدوات قائمة على البيانات، أن تستفيد بشكل كبير من هذه الشبكات حيث توفر وصولاً إلى معلومات أكثر تفصيلاً. في الواقع، توفر شبكات المعرفة مجموعة واسعة من البيانات المترابطة والغنية بالمعنى، وتوفر رؤى قيمة حول تفضيلات المستخدمين وميزات العناصر. التحدي الرئيسي لدمج شبكات المعرفة في أنظمة التوصية هي كيفية تصميم تمثيل فعال للمعلومات الدلالية المضمنة ضمن اتصالات عالية التعقيد والتي تتطلب تخطيطًا دقيقًا لضمان أن النموذج يمكنه التقاط واستغلال واستخدام المعلومات ذات الصلة بطريقة ذات مغزى. علاوة على ذلك، فإن دمج شبكات المعرفة يطرح تحديا آخر: التعامل مع الأنماط المعقدة الموجودة في الرسم البياني. تتمحور هذه الأطروحة في مفترق طرق مجالين بحثيين: أنظمة التوصية والبيانات الدلالية. نبرز كيف يمكن استغلال التطورات الأخيرة في تقنيات التعلم الآلي للتعلم من شبكات المعرفة لإنشاء أنظمة التوصية. في إسهامنا الأول، نقترح تمثيلًا للمعرفة يتوافق مع شبكات المعرفة و شبكة المستخدم. يشمل هذا التمثيل تحويل الإتصالات إلى عقد، مما يسمح لنا بالحفاظ على الدلالة التي تحملها الخصائص أثناء مرحلة التعلم. لتمكين التوصيات، نطبق خوارزمية Node2vec، لتعلم تضمينات مشتركة للإتصالات والخصائص الدلالية. من خلال دمج الخصائص في عملية التضمين، نكتسب فهمًا أعمق لتفضيلات المستخدم ونواجه مشكلة تكرار التوصيات. تظهر النتائج تحسينًا في الأداء من حيث الدقة والتنوع. يُعرف إسهامنا الثاني بـ "Explain-KGCN"، وهو نظام توصية قابل للشرح والمدرك للمعرفة .أولاً، نصمم تمثيل معرفة بناءً على مقياس وزني تفاعلي جديد مصمم خصيصًا للبيانات المرتبطة. يدمج هذا المقياس المعلومات السابقة حول أهمية العقد في عن طريق تعيين أوزان شخصية لكل عقدة. نستفيد أيضًا من معلومات أكثر غنى حول تفضيلات المستخدم من خلال النظر في التغذية الراجعة الإيجابية والسلبية الواضحة من قبل المستخدم. بالإضافة إلى ذلك، نطور خوارزمية خاصة بالخاصية بناءً على شبكة Graph Convolutional Network (GCN)، وهي تقنية تعلم عميق قوية لتضمين الرسم البياني. المكون الأساسي في خوارزميتنا هو وظيفة تجميع تضمين GCN محددة للعلاقة، والتي تستخرج تمثيلات منخفضة الأبعاد لكل عقدة من خلال جمع البيانات من الجيران ذوي العلاقة المحددة بالعلاقة. يمكننا بهذا النهج نمذجة دلالة العلاقات المختلفة بشكل أكثر وضوحًا، مما يؤدي إلى تضمينات ذات جودة أفضل وتوقعات أكثر دقة وتفسيرات دقيقة. بناءً على المساهمات السابقة، يتمتع إسهامنا الثالث بتوحيد عملية التوزيع باستخدام شبكات الانتباه البياني Graph Attention Networks (GATs). نستكشف إمكانات البيانات المرتبطة للتوصية القابلة للتفسير باستخدام GATs، بهدف استغلال المعلومات الدلالية بشكل كامل والحفاظ على المعرفة الأساسية المغلقة في العلاقات. من خلال دمج المعرفة الإضافية من سحابة LOD واستخدامGATs، نقوم بتوليد تمثيلات فيكتورية لكل عقدة في الرسم البياني. النهج المقترح يحسن الأداء التنبؤي من حيث الدقة والانتباه والتنوع، مستغلاً البيانات المهيكلة الغنية لتقديم تفسيرات للتوصيات عبر ثلاثة مجالات مختلفة.

الكلمات الدلالية. البيانات المرتبطة، تضمين الرسوم البيانية، الرسم البياني المعرفي، الويب الدلالي، أنظمة التوصية، القابلية للتفسير، التنوع.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Definition
AI	Artificial Intelligence
AUC	Area Under the Curve
CBF	Content-Based Filtering
CF	Collaborative Filtering
CNN	Convolutional Neural Network
DC_r	Degree Centrality-Relational
DL	Deep Learning
Explain-KGCN	Explainable Knowledge Graph Convolutional Networks
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GNN	Graph Neural Network
GraRep	Graph-Regularized Embedding Propagation
GRL	Graph Representation Learning
HIN	Heterogeneous Information Network
HTTP	Hypertext Transfer Protocol
I-KNN	Item-based K-Nearest Neighborhood
ILD	Intra-list diversity
KB	Knowledge Base
KG	Knowledge Graph
KGE	Knowledge Graph Embedding
kNN	k-Nearest Neighbors
LD	Linked Data
LOD	Linked Open Data
MAE	Mean Absolute Error
MDS	Multi-Dimensional Scaling
ML	Machine Learning
NLP	Natural Language Processing
ODR	Ontology of Dietary Recommendation
OWL	Web Ontology Language
PCA	Principal Component Analysis
RDF	Resource Description Framework
ReLu	Rectified Linear Unit
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RS	Recommender System
SPARQL	SPARQL Protocol and RDF Query Language
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TF-IDF	Term frequency-inverse document frequency
URI	Uniform Resource Identifier
VSM	Vector Space Model
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

Scientific Productions

Boughareb, R., Seridi, H., & Beldjoudi, S. (2019). A knowledge graph-based recommender system. In Proceedings of the 8th International conference on innovation and new trends in information technology (INTIS'19) (pp. 27-32). Tangier, Morocco.

Boughareb, R., Seridi, H., & Beldjoudi, S. (2023). Explainable recommendation based on weighted knowledge graphs and graph convolutional networks. Journal of Information and Knowledge Management, 22(3). doi: 10.1142/S0219649222500988.

Boughareb, R., Seridi, H., & Beldjoudi, S. (To be published). Improving explainability and diversity in recommender systems with the use of graph attention networks in joint entity-relation representation. Journal of Web Engineering.

Boughareb, D., Khobizi, A., Boughareb, R., Farah, N., & Seridi, H. (2019). Tag Recommendation for limited access papers annotation. In Proceedings of the 8th International conference on innovation and new trends in information technology (INTIS'19) (pp. 245-251). Tangier, Morocco.

Boughareb, D., Khobizi, A., Boughareb, R., Farah, N., & Seridi, H. (2020). A graph-based tag recommendation for just abstracted scientific articles tagging. International Journal of Cooperative Information Systems, 29(3). doi: 10.1142/S0218843020500045.

General Introduction

Contents

1.1 Context of the thesis: Inefficient use of Semantic Data

Semantic Web. The World Wide Web's (WWW) expansion has significantly changed how people produce and consume information. People all over the world may now access a broad array of information at their fingertips because to the widely available internet connectivity. As a result, users have become increasingly empowered to generate their own content.

The concept of the Semantic Web, often referred to the Web of Data, takes this evolution one step further by moving from a Web of interconnected Web pages, which are mainly intended for human consumption, towards a more sophisticated Web of Linked Data that is both machine-readable and human-understandable. The Semantic Web is a collection of standards and protocols that aims to make it possible to represent and process knowledge on the Web. In order to represent and describe data, it relies on a number of important technologies, including RDF (Resource Description Framework), RDFS (Resource Description Framework Schema), and OWL (Web Ontology Language), while SPARQL (SPARQL Protocol and RDF Query Language) is used to query RDF data.

At present, the Web of Data is home to an extensive collection of RDF data that has unlocked a plethora of opportunities for research, innovation, and development. This data can be utilized to gain valuable insights, and make accurate predictions across multiple domains, including healthcare, education, finance, and more. Researchers and data scientists can use this data to perform advanced analytics, identify patterns, and create machine learning models that can automate decision-making processes. Furthermore, the Semantic Web enables data integration across various domains, allowing datasets from disparate sources to be merged to achieve a more complete and accurate comprehension of a particular subject.

Linked Data, unlocking the power of the Semantic Web. Linked Data (LD) refers to a set of best practices for publishing and linking structured data on the Web, which were originally proposed by Tim Berners-Lee, the mind behind the Web, and are commonly known as the Linked Data principles. At the core of these principles is the use of unique identifiers for resources, providing metadata about resources using standardized machine-readable vocabularies, and using hyperlinks to connect resources to each other.

The overarching goal of the Linked Data is to increase data's transparency and accessibility as a shared resource, while also ensuring that it is technically interoperable, legally compliant, and economically feasible for a wide range of use cases. By adhering to these guidelines, data can be published and interlinked across different sources and domains, creating a Web of data that is more easily queried, interoperable, and analyzed.

An example of a large-scale application of Linked Data principles is the Linked Open Data (LOD) initiative, where various organizations and individuals have published a significant amount of RDF data in publicly accessible datasets that are connected to form the LOD cloud. This cloud offers new opportunities for data integration, analysis, and knowledge discovery as it is a decentralized network of linked datasets that can be searched and studied by both machines and people.

Knowledge Graphs, expanding the horizons of the Semantic Web and unleashing the full potential of data. Knowledge Graphs (KGs) constitute the foundation of the LOD cloud and the new era in Knowledge Representation. By adopting a structured labeled graph format, KGs effectively capture and emphasize the intricate interconnections among various data points. Coined by Google in 2012, the term "Knowledge Graph" marked a pivotal moment when the tech giant introduced its groundbreaking graph-

structured knowledge as the fundamental framework for a novel Web search strategy. Departing from conventional word processing, this visionary approach embraced a symbolic representation of knowledge, encapsulating the essence of their mission: "*we deal with objects, not strings.*"

In recent years, the popularity of KGs has surged significantly. This rising trend can be attributed to the recognition of the immense benefits that graph technologies offer in understanding and leveraging data effectively.

One key advantage of KGs is their ability to capture and represent complex relationships between entities and concepts. Traditional data models often struggle to capture the intricacies and interdependencies present in real-world data. Knowledge Graphs, on the other hand, excel in capturing these relationships by leveraging the graph data structure. This enables to gain a more holistic and interconnected view of data, uncovering hidden insights and patterns that were previously difficult to identify.

Another driving factor behind the popularity of KGs is their ability to empower advanced data analytics and reasoning. By representing knowledge in a structured and interconnected manner, they provide deeper understanding of data, enable powerful querying, advanced analytics, and reasoning capabilities.

Knowledge Graphs meet Recommender Systems. Recommender Systems (RSs) are software algorithms designed to suggest products, services, or content to users based on their preferences and behaviors. These systems use various techniques, such as collaborative filtering, content-based filtering, and hybrid filtering, to analyze data on user interactions and item characteristics to generate personalized recommendations. One real-world example of a RS is Netflix. Netflix employs a hybrid recommendation algorithm that analyzes user viewing history, ratings, and preferences to suggest personalized movies and TV shows.

Evaluating RSs extends beyond accuracy metrics and encompasses multiple dimensions that contribute to user satisfaction and enhance the recommendation experience. Alongside accuracy, dimensions such as diversity, explainability, novelty, serendipity, and user trust play pivotal roles in evaluating the effectiveness of RSs. As an example, diversity is a crucial aspect that ensures users are exposed to a broad range of items. By recommending diverse items, including those from different genres, categories, or niche interests, RSs foster serendipitous

discovery and prevent users from being confined to a limited set of recommendations. This dimension promotes user engagement, satisfaction, and a sense of exploration. Explainability is another vital dimension that empowers users to understand and trust the recommendations provided by the system. Providing explanations enhances user confidence, facilitates feedback, and enables users to refine their preferences, resulting in more personalized and accurate recommendations over time.

RSs, being data-driven tools, can greatly benefit from the use of KGs, as they give them access to more details about both users and items. In fact, KGs offer a wide range of interconnected and semantically-rich data, which can offer insightful information about user preferences and item features. This, in turn, can lead to higher user satisfaction.

The primary technical tracks on Knowledge Graph-based recommendation research field have been primarily focused on feeding RSs with rich structured data and applying classic Machine Learning (ML) algorithms commonly used in Information Retrieval (IR), such as Vector Space Model (VSM), Support Vector Machine (SVM), and Genetic Algorithms (GA). The major difficulty with early approaches is that traditional ML methods may struggle to effectively handle the complex patterns within knowledge graphs (KGs), particularly in terms of scalability. KGs can be massive, containing millions or even billions of nodes and edges, which poses challenges for traditional ML algorithms in terms of efficient processing and inference. This limitation hampers the effectiveness of traditional ML methods and subsequently leads to reduced accuracy in KG-based RS. To overcome these challenges and improve RS accuracy, it is crucial to employ more effective methods able to handle KGs and their intricate patterns, enabling scalable and accurate recommendations.

Graph Embedding Methods, a promising solution. Graph Representation Learning (GRL) methods, also known as Graph Embedding methods, have emerged as powerful techniques that enable the representation of graph-structured data in a continuous vector space. These methods have demonstrated significant success in various graph analysis tasks, such as node classification, link prediction, and community detection.

One of the key advantages of GRL methods is their ability to tackle the challenge of scalability. Traditional ML algorithms often struggle with the computational demands imposed by large-scale graphs, limiting their applicability in real-world scenarios. However, GRL methods have introduced scalable solutions based on Neural Network architectures and

optimization techniques that can efficiently handle big graphs.

In the context of RSs, GRL methods have proven particularly valuable. By learning representations that encode the structural properties of the user-item graph, GRL methods enable more accurate and personalized recommendations, enhancing the user experience and improving the quality of recommendations. However, it is important to note that while GRL methods employ nonlinear mapping functions, they might not be explicitly designed to analyze semantic graphs. Semantic graphs are characterized by highly meaningful information and often contain rich, structured knowledge that goes beyond basic node and edges. Consequently, the complexity and depth of semantic graphs may present challenges for GRL methods in capturing their nuanced semantics and extracting comprehensive insights.

Deep Graph Embeddings for more efficient Knowledge Graphs processing and analysis. With the proliferation of mobile devices, people have greater accessibility to the internet and social media platforms, allowing them to express their opinions, preferences, and experiences more readily. This accessibility has led to an exponential growth in user-generated content, encompassing a vast amount of complex data, which can be challenging to analyze and comprehend effectively.

In the context of KG-based RS, to make sense of this complex data and provide meaningful recommendations, it becomes necessary to employ more effective methods. These methods should be compatible with the new levels of complexity and scalability of knowledge graphs (KGs) and the intricate patterns they contain.

One promising approach to address this challenge is the use of Deep Graph Representation Learning (DGRL), especially Graph Neural Networks (GNNs). While not specifically designed for KGs, these methods have shown great promise in learning from complex graph structures. Their deep neural architectures and techniques enable the extraction of meaningful representations from complex data, capturing intricate patterns and relationships.

How is the use of Deep Graph Representation Learning in KG-based RSs can flawed? Although Deep Graph Representation Learning provides a better approach to handle the complex patterns hidden in scalable heterogeneous semantic data, there are still some outstanding challenges that need to be addressed to fully realize the potential of this data. In fact, insufficient attention has been paid to resolving the semantic loss problem, which

ultimately results in a lack of interpretability, making it difficult to understand how the embeddings are learned and how they contribute to the recommendation process. This problem is due to the non-linear transformations that occur across layers, which play a fundamental role in extracting complex patterns from the data.

Preserving semantics in KGs is essential for accurate analysis and reliable recommendations, not only in terms of accuracy, but also it can bring other benefits like diversity and explainability.

Effective Knowledge Representation, Overcoming semantics loss challenge. The design of a KG-based RS poses notable challenges, with one of the most critical being the effective representation of knowledge. This challenge encompasses incorporating the user-item bipartite graph and leveraging the complex semantic relationships embedded within the KG. An effective representation should successfully capture the semantic relationships among entities within the KG while also considering the preferences and interests of users, which play a pivotal role in personalized Recommendation Systems.

This Ph.D. thesis lies at the crossroads of two research domains, Recommender Systems and Semantics, showing how the latest advancements in Machine Learning can be used to learn features from Knowledge Graphs for recommendation purposes.

1.2 Research Challenges

Our study focuses on three primary challenges in recommendation using KGs:

- **Challenge I** is effectively leveraging the semantic relationships within KGs to improve recommendation.

- **Challenge II** is designing a recommendation model that fully exploits the semantic information behind the high-order connections.

- **Challenge III** is using KGs to enhance RSs, extending the focus beyond accuracy and encompassing additional dimensions such as diversity and explainability.

1.3 Research Questions and Contributions

The main research question investigated in this thesis is:

How can a good representation of knowledge lead to an effective semantic preservation which generates recommendations that not only enhance precision but also bring enhancements in areas such as diversity and explainability?

For a more detailed analysis, we present the specific research questions that we studied:

RQ I: *How to incorporate the knowledge encoded in KGs into the user-item graph, while retaining the semantics conveyed by the predicates during the learning phase, with the goal of enhancing recommendation precision and promoting diversity in the recommended items?*

Hypothesis I: By suggesting a Knowledge Representation that permits the simultaneous learning of low-dimensional embeddings for both entities and properties, it is possible to prioritize the maintenance of semantic information linked with attributes and produce recommendations that are more correct.

Contribution I: *We propose a Knowledge Representation that aligns with both Knowledge Graph and the user-item bipartite graph. This representation involves converting properties (edges) into entities (nodes) allowing us to preserve the semantics conveyed by the predicates during the learning phase. To enable recommendations, we apply a Graph Representation Learning algorithm to jointly learn embeddings for entities and semantic properties.*

The first step toward attaining further recommendation advancements is the efficient representation of semantics to enable accurate recommendations. Within this challenge, we explored many sub-research questions, including:

- *How can a Recommender System leverage the efficacity of the GRL algorithm node2vec in learning features from graphs?*
- *How can translational methods generate item recommendations?*
- *Is the proposed Knowledge Representation can insure the diversity of recommendations?*

RQ II: *Given the difficulties in retaining semantic information in "black box" models, how can the successful usage of Knowledge Graphs in conjunction with Deep Learning-based Graph Embedding approaches be achieved for Recommendation Systems?*

Hypothesis II: Proposing a weighted Knowledge Representation in which nodes are weighted according to their importance in user profiles, and learning embeddings for entities based on their semantic relationships, can improve the accuracy of personalized recommendations and provide explainability. Specifically, the weighted representation will enable better capturing of user preferences and interests, while the use of relation-based embeddings will facilitate the identification of relevant entities for each user and preserve semantics.

Contribution II: *We propose Explain-KGCN, an Explainable KG-aware Recommender System. First, we design a weighted Knowledge Representation based on a novel relational weighting metric which is designed mainly to deal with KGs. This metric incorporates prior information about the importance of nodes in the KG by assigning personalized weights for each node and benefits from richer information about all possible user preferences by jointly considering both positive and negative explicit user feedback. Also, we develop a property-specific algorithm based on a Graph Convolutional Network (GCN), a robust Deep Learning Technique for Graph Embedding. The core component of the proposed algorithm is a relation-specific GCN aggregator function that extracts low-dimensional representations for each node in the Knowledge Graph by gathering data from its relation-specific neighbors. This approach helps the system to model the semantic of different relations more explicitly, which leads to better quality embeddings, more accurate predictions, and precise explanations.*

To achieve further improvements in recommendations, the initial step is to have a reliable representation of semantics that allows for precise recommendations. In order to tackle this challenge, we investigated several sub-research questions:

- *How GCNs could be leveraged to generate recommendations that are not only precise but also interpretable and attuned to semantics, based on both KGs and user-item interactions?*

- *How negative and positive users' feedbacks affect the prediction performances?*

- *How does the proposed metric contribute to preserving semantics and enhancing the*

system accuracy?

- *Is the proposed Knowledge Graph can insure the explainability of recommendations?*

RQ III: *Considering the challenges posed by the complexity and black box nature of Deep Graph Learning, when confronted with a knowledge representation that prioritizes semantic preservation, how exactly does Deep Graph Learning facilitate the promotion of such preservation? Furthermore, how does this promotion result in not only more accurate and diverse recommendations, but also providing explanations regarding the diversity of these recommendations?*

Hypothesis III: By integrating feature extraction and deep-learning-based node importance computing within the Deep Graph Learning framework, we can effectively address the challenges of complexity and black box nature. This integrated approach, where the learning embeddings phase and node importance computing are not separated, is expected to promote semantic preservation while harnessing the superior effectiveness of deep graph learning as a machine learning algorithm. By leveraging this integrated solution, we anticipate achieving improved performance in terms of accuracy, diversity of recommendations, and the provision of comprehensive explanations regarding the diversity observed in these recommendations.

Contribution III: *We use a Knowledge Representation approach that prioritizes semantic preservation, utilizing the wealth of background knowledge available from the LOD cloud. Our representation facilitates the joint learning of both semantic properties and entities, enabling a comprehensive understanding of the underlying knowledge. To enhance the efficiency of the graph attention Network (GAT) algorithm, a powerful Deep Graph Learning (DGL) algorithm, we propose a novel semantic-based information gain property selection method. This method effectively reduces the complexity of the GAT algorithm while capturing more relevant semantic relationships between entities and properties. By leveraging this property selection technique, our model achieves improved performance in terms of accuracy and diversity of recommendations. The resulting recommendations reflect a better understanding of the underlying semantics, thus offering more valuable insights to the end-users.*

Within the context of the aforementioned challenges, we explored several sub-research questions, including:

- *How can a Recommender System leverage the efficacity of the DGRL algorithm GAT in*

learning features from graphs?

- *How can the property extraction method Information Gain (IG) help to reduce GAT complexity?*
- *Can the proposed Recommender System ensure the diversity of recommendations?*
- *Can the proposed Recommender System provide an explanation for the diversity of recommendations?*

1.4 Organization of the thesis

The thesis includes five chapters and it is organized as follows:

General Introduction: It provides an overview of the research topic, outlines the motivation for conducting the study, presents the research questions and hypotheses that will guide the investigation, and sets the scope of the study.

Chapter I entitled Knowledge Representation: This chapter offers an overview of Knowledge Representation, highlighting the essential concepts and technologies that underpin the Semantic Web. Additionally, it helps to differentiate commonly confused terms such as Linked Data, Linked Open Data, and Knowledge Graphs, thus contributing to a more complete comprehension of this expansive field of research.

Chapter II, entitled "Recommender Systems Meet Knowledge Graph", provides a comprehensive overview of different approaches to recommendation systems and the problems they confront. Moreover, it discusses the state-of-the-art Semantic-aware Recommender Systems and presents a comparative analysis of the different approaches, highlighting their respective strengths and weaknesses.

Chapter III, entitled "From Knowledge Graphs to Personalized Recommendations: Leveraging Graph Embeddings For Improved Accuracy and Diversity", describes the first contribution of the thesis, in which we propose a design of a Knowledge Graph that aligns with both Linked Data and the user-item bipartite graph, with

the aim of preserving semantics while simultaneously enhancing recommendation accuracy and diversity. The proposed approach employs a Graph Learning algorithm to learn embeddings and generate recommendation predictions.

Chapter IV entitled Explain-KGCN: "A Convolutional Weighted Knowledge Graph for Explainable Recommendation," describes the second contribution of the thesis, in which we propose Explain-KGCN, an Explainable GCN-based Knowledge-aware Recommender System. The proposed approach includes the design of a semantically-preserving weighted Knowledge Representation, the use of the Deep Representation Learning algorithm GCN to learn item embeddings and generate explainable recommendations.

Chapter V, entitled "Improving Explainability and Diversity in Recommender Systems with Use of Graph Attention Networks in Joint Entity-Relation Representation", describes the third contribution of the thesis. In this chapter, we propose a semantically-preserving KG-based recommendation approach that leverages Graph Attention Networks (GATs). The proposed approach includes a semantic-based Information Gain property selection phase, which aims to reduce the complexity of the GAT learning process. We utilize the Deep Representation Learning algorithm GAT to learn item embeddings and generate diverse and explainable recommendations.

General Conclusion: This thesis summarizes the main findings of the study and identifies potential avenues for future research. It also acknowledges the limitations of the study and suggests areas for further exploration.

Chapter 1

Knowledge Representation

Contents

Introduction

The ability to understand, reason, and interpret knowledge is a natural trait in humans. Similarly, computer science and Artificial Intelligence have long aimed to replicate this ability in machines through knowledge representation. It is worth noting that knowledge representation goes beyond storing data in a database but also lies in being able to learn and improve on this knowledge, much like how humans adapt to new information.

The Semantic Web, which aims to enhance the current Web's interoperability and intelligibility, relies on the concept of knowledge representation as a fundamental building block. In fact, one of the main goals of Semantic Web is to enhance the representation of information on the current Web in a way that enables machines to interpret it like humans do. This enables advanced reasoning, search, and analysis capabilities, leading to more efficient and accurate knowledge management.

The development of the Semantic Web can be traced back to three distinct phases. The initial phase focused on incorporating Knowledge Representation into Web standards. In the second phase, the emphasis was on data management, Linked Data, and potential applications. The current third phase, which has been marked by the widespread adoption of the Web by a variety of devices and things, has put a renewed emphasis on Knowledge while utilizing Knowledge Graphs.

This chapter aims to offer a thorough overview of Knowledge Representation, focusing on the key concepts and technologies that underpin the Semantic Web including Linked Data, Linked Open Data, and Knowledge Graphs. Additionally, an exploration of the applications, limits, and open research issues of Knowledge Graphs will be undertaken.

1.1 A little bit of history

The inception of the Semantic Web dates back to 1994 when Tim Berners-Lee, the innovator behind the World Wide Web (WWW), presented his vision for the Semantic Web at the First International WWW Conference (Berners-Lee *et al.*, 1994).

Six years later, Berners-Lee and his colleagues (Berners-Lee *et al.*, 2001) disseminated the term Semantic Web with their vision of a more expansive and improved version of the

traditional Web: to create a genuinely valuable storehouse of information on the Web, two essential elements were necessary. Firstly, the inclusion of metadata about the information available on the Web, that could be read and understood by machines. Secondly, the addition of values to relationship hyperlinks that computers could use to guide searches.

In 2006, the World Wide Web Consortium (W3C) published a set of Semantic Web recommendations that provide a framework for creating, consuming, and publishing, structured machine-readable data (Shadbolt *et al.*, 2006). These recommendations encompassed a set of technologies, vocabularies, and frameworks for Knowledge Representation.

Today, the Semantic Web is being used in a wide range of applications, including search engines, e-commerce, scientific research, healthcare, and social networking. The vision of a fully realized Semantic Web is still a work in progress, but the potential benefits for both humans and machines are immense.

1.2 Semantic Web Definition

In his book "Weaving the Web" (Berners-Lee, 2000), Tim Berners-Lee, describes the Semantic Web as:

> *"The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation."*

According to this definition, the Semantic Web provides a way to represent knowledge in a standardized and structured format, enabling machines to more easily process and reason about it. This has the potential to revolutionize many areas, including data integration (Gardner, 2005; Langegger, 2005; Goble and Stevens, 2008) search (Madhu *et al.*, 2011, Patel and Jain, 2021), and analytics (Khodadadi and Sinnott, 2017; Piselli *et al.*, 2022).

The exchange and integration of Web data is made possible through a collection of technologies, tools and standards that function as the essential building blocks of a system that can support the development of a semantically-enriched Web.

1.3 Semantic Web Technologies

Semantic Web technologies refer to a set of standards, tools, and methodologies designed to enable the exchange and integration of data on the Web in a machine-readable format. They are organized as a conceptual layered architecture, where each layer represents a specific technology and builds upon the layers beneath it. The layers include the XML layer, the RDF layer, the Ontology layer, and the Logic layer, among others. Figure 1.1 illustrates the typical representation of the Semantic Web technologies as introduced by the W3C.

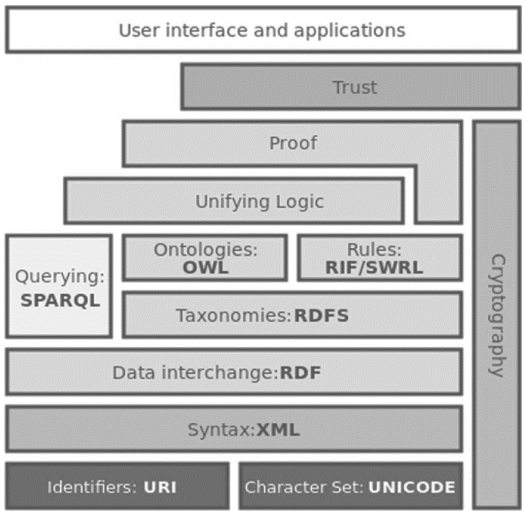


Figure 1.1. The W3C's Semantic Web layered architecture (Ahmed *et al.*, 2010**).**

The architecture provides a structured approach to designing and implementing Semantic Web applications, allowing for greater interoperability and ease of development. The following subsections delve into the most important technologies in greater detail. We note that in this thesis, we consider Linked Data, Linked Open Data, And Knowledge Graphs as integral technologies. Their inclusion is justified by the numerous benefits they bring to Semantic Web applications, such as improved interoperability and streamlined development processes.

1.3.1 Uniform Resource Identifier

Uniform Resource Identifier (URI) is the foundation layer of the Semantic Web architecture. It provides a standardized way to uniquely identify resources on the Web, and allow them to be accessed by machines and humans alike. They are made up of a scheme (such as http or https), a domain name, and a path to the resource. URIs can also include query parameters and fragment identifiers.

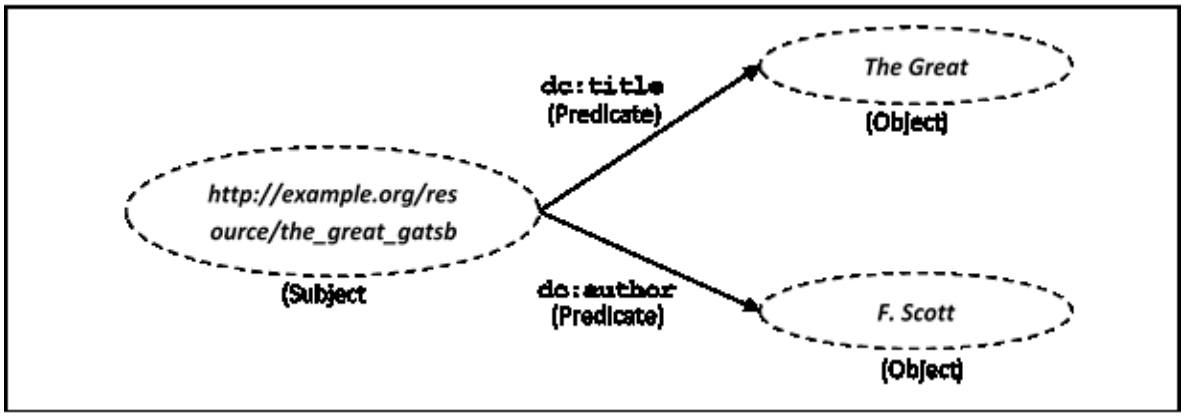


Figure 1.2. The visualization of RDF triples.

For example, the URI *'https://www.example.com/resource/the_great_gatsby?id=1#reviews'* identifies *'The Great Gatsby'* resource book on the website *'www.example.com'*. The scheme is *https*, and the path is *'/resource/the_great_gatsby'*. The query parameter specifies the ID of the laptop, while the fragment identifier specifies the section of the page where the book reviews are located.

1.3.2 Resource Description Framework

The Resource Description Framework (RDF) (Lassila and Swick, 1999) is a standard model for representing and exchanging information on the Web in a structured manner. It offers a means of describing diverse resources, including people, places, and things, and their interrelationships. The RDF model manages data by making statements about resources in expressions of three parts: a subject, a predicate, and an object referred to as "triple", where:

- **The Subject (σ)** represents any resource described by a predicate, and that can be referenced by an Internationalized Resource Identifiers (URI).

- **The Object (o)** represents either the value of the predicate associated with a specific resource, or the Subject of another triplet, forming in this case a directed labeled graph.

- **The predicate (ρ)** describes the relationship between the subject and the object.

In a triple-based representation, each predicate is represented as a directed edge that connects the subject node of the triple to its corresponding object node. By visually representing RDF triples in this way, complex relationships between different resources can be easily understood and analyzed. This graphical approach also facilitates exploration of large and complex

datasets, enabling identification of patterns and anomalies in the data.

While RDF data has a unique graphical triple format, it can also be expressed in various syntaxes. The first syntax for RDF was based on XML, but more intuitive syntaxes such as Turtle, N-Triples, JSON-LD, and RDFa are now commonly used. Table 1.1 and Figure 1.2 provide an illustrative example of RDF triple statements that have been encoded in XML format, along with their respective visualizations.

Table 1.1. An example of RDF data represented in the XML format.

Xml

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/"

<rdf:Description
rdf:about="http://example.org/resource/the_great_gatsby">
    <dc:title>The Great Gatsby</dc:title>
    <dc:author>F. Scott Fitzgerald</dc:author>
</rdf:Description>

</rdf:RDF>
```

This RDF/XML example describes a book resource with a title and an author using RDF syntax. The subject of the triple is the book resource with a URI of *'http://example.org/resource/the_great_gatsby'*, and the predicates are the title *'The Great Gatsby'* and author *'F. Scott Fitzgerald'* predicates of the book.

In summary, RDF offers a flexible approach to Knowledge Representation, enabling resources to be described, reused, and shared across different applications. However, RDF lacks the ability to define the semantics of manipulated data, nor does it provide a mechanism for imposing type constraints on resources. Thus, it is essential to provide agents that read these resources with the means to understand, for instance, that the URI *'http://example.org/resource/F_Scott_Fitzgerald'* represents the category *'person'*, which can possess various properties and be linked to other concepts. To achieve this, vocabularies are required to model this semantics.

1.3.3 Resource Description Framework Schema

The Resource Description Framework Schema (RDFS) (Brickley and Guha, 2004) is a meta-model for defining RDF vocabularies or schemas. It extends the syntax of RDF triples with new

metadata to define the notions of:

- **Classes:** which are used to group resources with similar characteristics.

- **Properties:** which are used to describe the relationships between resources.

- **Subclassing and subproperty relationships:** which allow the creation of more specific classes and properties by inheriting characteristics from more general ones.

- **Domain and range:** which define the type of resources that can be subjects and objects of a property, respectively.

- **Datatypes:** which define the data types of property values, such as integers, strings, or dates.

Table 1.2 presents the updated version of the previous example RDF triple statements encoded in XML format, which have been modified from RDF to RDFS.

Table 1.2. An example of RDFS statements represented in the XML format.

Xml

```
<?xml version="1.0"?>
<rdfs:RDF xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
          xmlns:dc="http://purl.org/dc/elements/1.1/"

<rdf:Description
rdf:about="http://example.org/resource/the_great_gatsby">
 <dc:title
     rdf:datatype="http://www.w3.org/2001/XMLSchema#string">The Great
Gatsby
 </dc:title>
 <dc:author
     rdf:resource="http://example.org/resource/f_scott_fitzgerald"/>
</rdf:Description>


<rdf:Description
rdf:about="http://example.org/resource/f_scott_fitzgerald">
 <dc:title>F. Scott Fitzgerald</dc:title>
 <rdfs:subClassOf rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
</rdf:Description>

</rdfs:RDF>
```

In this modified example, *'rdf:datatype'* attribute has been added to specify the data type of the literal value for the *'dc:title'* property. Additionally, a *'rdfs:subClassOf'* property has been added to indicate that this resource is a subclass of *'Person'*.

RDFS provides a richer vocabulary than basic RDF for designing a better Knowledge Representation model, and enables the creation of more expressive and reusable RDF vocabularies. However, it is not enough to define a hierarchy of classes or properties to constitute an effective semantic representation of resources. Indeed, RDFS has no built-in mechanism for automated reasoning on knowledge patterns, such as class combination, class equivalence, or cardinality restrictions. As a result, RDFS is limited in its ability to express complex domain knowledge and infer new knowledge from existing data. To address these limitations, more expressive Knowledge Representation languages and reasoning engines, such as OWL, have been developed on top of RDFS.

1.3.4 Web Ontology Language

The Web Ontology Language (OWL) (Bechhofer *et al.*, 2004) was designed to enrich the RDFS framework by defining a more complete vocabulary. This richness is materialized by the addition of new notions such as:

- **Class or property equivalence:** to specify that two or more classes (or properties) are equivalent, for example, the *'Movie'* and *'Film'* classes represent the same concept.

- **Cardinality:** refers to the number of values that a property can take. For example, for an instance of the *'User'* class, a property like *'Rated'* may have a cardinality of zero or more, indicating that the user may not have rated any movies, or may have rated multiple movies.

- **Symmetry:** indicates that if a property relates an instance A to an instance B, it also relates instance B to instance A.

- **Transitivity:** indicates that if a property relates an instance A to an instance B, and the property also relates instance B to instance C, then the property also relates instance A to instance C.

Table 1.3 defines an OWL ontology for books, which includes three classes: *'Person'*, *'Author'*, and *'Book'*. The *'Person'* and *'Book'* classes are subclasses of *'Resource'*, while *'Author'* is a subclass of *'Person'*. In addition to the classes, the ontology defines two properties: *'title'* and *'hasAuthor'*. *'title'* is a datatype property that relates a book to its title, which is represented as a string. *'hasAuthor'* is an object property that relates a book to its author, which must be an instance of the *'Person'* class. The example also includes two named

individuals: *'the_great_gatsby'* and *'f_scott_fitzgerald. the_great_gatsby'* is an instance of the *'Book'* class and has a *'title'* property that is set to *'The Great Gatsby'*.

Table 1.3. An example of an OWL ontology.

Xml

```
<owl:Ontology rdf:about="http://example.org/myontology"/>

<owl:Class rdf:about="http://example.org/myontology/Person">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Resource"/>
</owl:Class>

<owl:Class rdf:about="http://example.org/myontology/Author">
  <rdfs:subClassOf rdf:resource="#Person"/>
</owl:Class>

<owl:DatatypeProperty rdf:about="http://example.org/myontology/title">
  <rdfs:domain rdf:resource="http://example.org/myontology/Book"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>

<owl:ObjectProperty rdf:about="http://example.org/myontology/hasAuthor">
  <rdfs:domain rdf:resource="http://example.org/myontology/Book"/>
  <rdfs:range rdf:resource="http://example.org/myontology/Person"/>
</owl:ObjectProperty>

<owl:Class rdf:about="http://example.org/myontology/Book">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Resource"/>
</owl:Class>

<owl:NamedIndividual
rdf:about="http://example.org/resource/the_great_gatsby">
  <rdf:type rdf:resource="http://example.org/myontology/Book"/>
  <my:title rdf:datatype="http://www.w3.org/2001/XMLSchema#string">The
Great Gatsby</my:title>
  <my:hasAuthor
rdf:resource="http://example.org/resource/f_scott_fitzgerald"/>
</owl:NamedIndividual>

<owl:NamedIndividual
rdf:about="http://example.org/resource/f_scott_fitzgerald">
  <rdf:type rdf:resource="http://example.org/myontology/Person"/>
  <my:title rdf:datatype="http://www.w3.org/2001/XMLSchema#string">F.
Scott Fitzgerald</my:title>
</owl:NamedIndividual>
```

1.3.5 SPARQL Protocol and RDF Query Language

To achieve the objectives of the Semantic Web, the W3C has created a suite of technologies, which includes resource description languages such as RDF, RDFS, and OWL, as well as query protocols that are optimized for working with knowledge. SPARQL (Prud'hommeaux and

Seaborne, 2008) is a standardized query language for RDF graphs, leveraging a graph-based approach to identify paths and retrieve query results. With its versatile syntax and robust querying capabilities, SPARQL is a valuable tool for exploring and discovering knowledge. The structure of a SPARQL query consists of the following elements:

- PREFIX declaration: this is optional and allows the query to use a namespace prefix to shorten URIs in the rest of the query.

- The operator: SPARQL queries consist of different operators that define the type of query to be formulated. The SELECT operator is a required keyword that specifies what kind of results should be returned. It can be used to select variables, literals, or URIs. The ASK operator is used to answer a query by identifying whether the searched pattern is present in the queried graph, returning a boolean result of 'yes' or 'no'. The CONSTRUCT operator is used to transform an RDF graph into another graph. Finally, the DESCRIBE operator returns a description of the resource passed as an argument in the form of an RDF graph.

- FROM attribute: it allows the interrogation of several RDF documents.

- WHERE attribute: this is a required keyword that specifies the pattern to match in the RDF graph. It consists of a set of triple patterns or other graph patterns, including optional patterns, filter expressions, and graph unions.

Table 1.4 provides the SPARQL query that retrieves the title and author of the resource with the URI *'http://example.org/resource/the_great_gatsby'* using the *'dc:title'* and *'dc:author'* properties. The "PREFIX" keyword is used to define a shorthand abbreviation for the namespace, making the query more concise. The "WHERE" clause specifies the pattern to match, which consists of two triples connected by the "AND" logical operator. The result would be a table with one row and two columns, containing the value of title and author for the resource.

Table 1.4. An example of a SPARQL query.

Ruby

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT ?title ?author
WHERE {
  <http://example.org/resource/the_great_gatsby> dc:title ?title .
  <http://example.org/resource/the_great_gatsby> dc:author ?author .
}
```

Result	
Title	**Author**
"The Great Gatsby"	http://example.org/resource/f_scott_fitzgerald

1.3.6 Linked Data

Linked Data (LD) (Berners-Lee, 2006) is a core pillar of the Semantic Web, it provides a set of best practices for creating links between semantic datasets. These principles, proposed by the Semantic Web Education and W3C's Outreach in 2007, serve as guidelines for sharing interlinked data in a machine-readable format on the Web, enabling efficient data processing and exchange.

To outline the roadmap towards this vision of the Web, Tim Berners-Lee described four LD principles:

a. Use URIs as identifiers for resources: all resources should be identified using a unique URI, which allow them to be unambiguously referenced and linked to other resources.

b. Use HTTP URIs so that people can look up these names: the URIs used should follow the Hypertext Transfer Protocol (HTTP) in order to dereference the different resources, so that when a user or machine agent follows the link, it can retrieve more information about the resource.

c. Use RDF to represent the data: to enable flexible and extensible data representation, should be represented using RDF.

d. Include links to other URIs to discover more resources: LD resources should contain links to other resources and datasets, creating a Web of interconnected data.

The Linked Open Data (LOD) initiative is a large-scale implementation of the Linked Data principles. Numerous organizations and individuals have published a vast amount of RDF data under an open license that aims to achieve greater transparency by making data a common good and accessible to all, while creating a technical, legal, and economic benefits for users. This data is distributed across multiple datasets, which are linked together to form a decentralized network of interconnected datasets that can be queried and analyzed by machines and humans alike. This network depicted in Figure 1.3 called LOD cloud, allows for more efficient and effective data integration, analysis, and knowledge discovery.

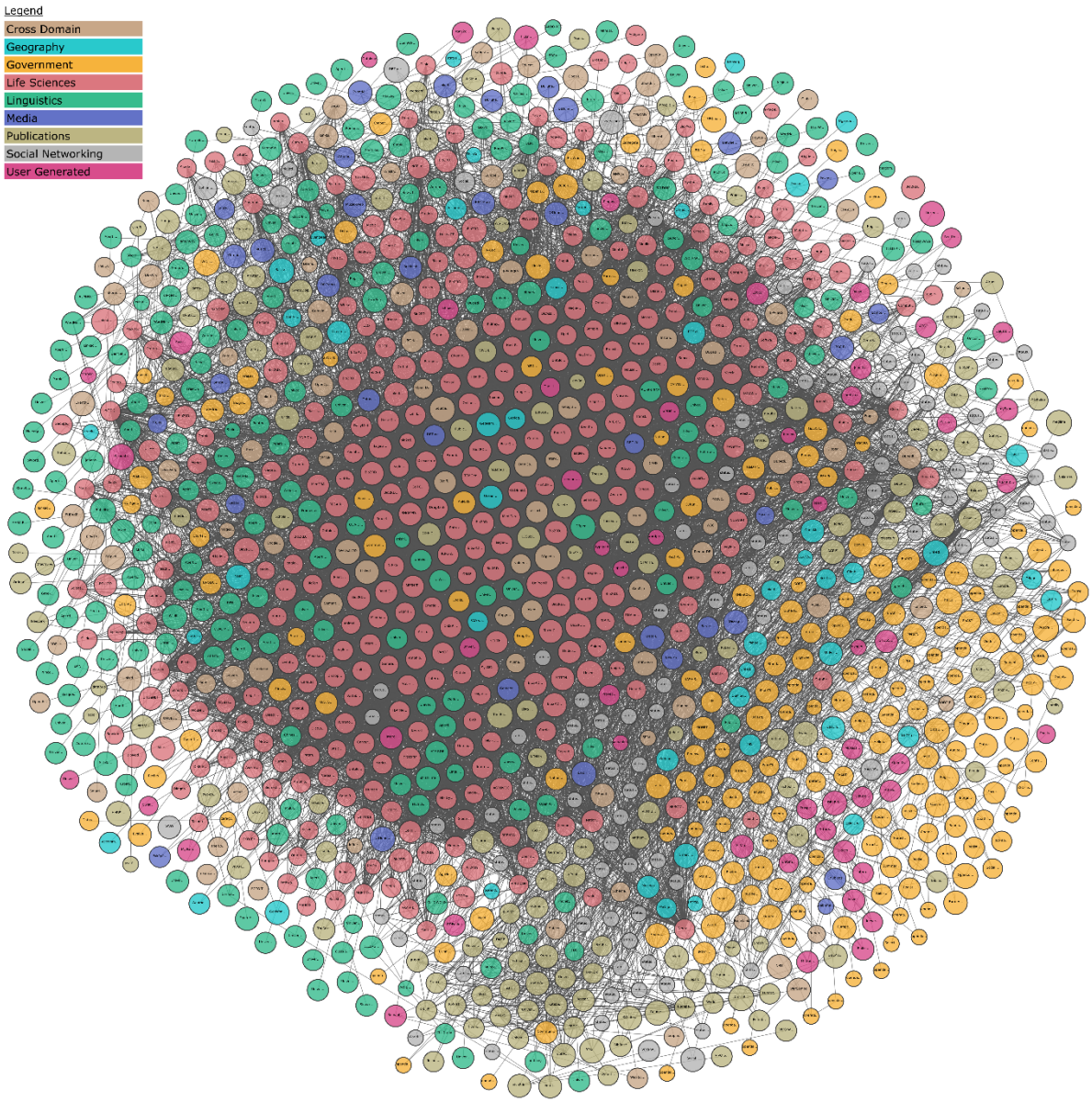


Figure 1.3. The Linked Open Data cloud in 2022-11-03 from lod-cloud.net.

In the LOD cloud illustration, datasets are represented as circles, and the links between them are depicted as lines. Each circle has a label indicating the name of the dataset, and the lines between them show the connections between the datasets. The size of the circle represents the size of the dataset, while the color of the circle indicates the domain of the dataset. For example, yellow circles represent datasets related to government information, while the turquoise blue circles represent datasets related to geographic information.

1.3.7 Knowledge Graphs

Knowledge Graphs (KGs) constitute the foundation of the LOD cloud and the new directions

for Knowledge Representation on the Semantic Web (Fensel *et al.*, 2020). They represent knowledge in a structured labeled graph format that highlights the relationships between data points. The term "Knowledge Graph", was invented by Google in 2012, when it introduced its graph structured knowledge as the backbone of a new Web search strategy (Hogan *et al.*, 2021). This strategy moved away from word processing to a more symbolic representation of knowledge, using the slogan "*we deal with objects, not strings*" (Ehrlinger and Wöß, 2016).

More formally, A KG is a labeled directed graph, where nodes N represent entities, edges represent relationships between entities E, and labels L on the edges capture the meaning of the relationships. It is a subset of the cross-product $N \times L \times N$.

1.3.7.1 Applications of Knowledge Graphs

Knowledge Graphs have become increasingly popular as more industries recognize the benefits of using graph technologies to understand their data (Yang and Liao, 2022). KGs are versatile and can be used in a wide range of domains, including:

- Healthcare: The healthcare industry faces the challenge of handling an enormous amount of diverse medical data, which are both a blessing and a curse. Fortunately, KGs are gaining popularity in the medical field, enabling researchers to integrate and structure medical data for various domains, map relationships between medical data, and discover new ways of using drugs to cure life-threatening diseases (Nicholson and Greene, 2020). Ontotext's[1] collaboration with AstraZeneca[2] on the LarKC project is a significant example of the company's contributions to the development of Knowledge Graph technology in the healthcare industry. The project's goal was to create a large Knowledge Graph that supports a holistic view of translational medicine, integrating diverse categories of information into a unified network of knowledge. The resulting graph allows for the exploration of causal relationships between objects that may not be immediately apparent in a single data source, making it a valuable tool for early hypothesis testing in life sciences. With the built-in ontology, the platform enables users to gain new insights into complex relationships between medical data, potentially leading to the

[1] https://www.ontotext.com/: is a technology company that specializes in creating semantic data management and knowledge discovery solutions for enterprises.

[2] https://www.astrazeneca.com/: is a biopharmaceutical company with a global presence that specializes in research, development, and marketing of medicines for some of the most critical illnesses worldwide.

development of new therapies and treatments for serious illnesses.

- Search engines: As the internet continues to expand, search engines must navigate through a vast ocean of information to provide accurate results for each search query. This is where Knowledge Graphs come in. Since Google's announcement of the first Knowledge Graph, they have become a crucial component of search engine navigation. The Google and Bing Knowledge Graphs are two of the most popular, enhancing modern search engines in several ways (Zou, 2020). Knowledge Graphs refine search results by creating rich connections between stored information, enabling search engines to create precise summaries relevant to the search query (Peng *et al.*, 2023). They also facilitate quick information retrieval, allowing users to delve deeper and find the most relevant information. In addition, Knowledge Graphs enable search engines to perceive entities and properties from natural language voice searches (Ji *et al.*, 2022).

- Entertainment: The entertainment industry, particularly social media, is undergoing a transformation with the help of Knowledge Graphs, which are being increasingly implemented alongside AI. These versatile technologies have numerous applications in media that enhance user satisfaction to new heights. One of the most significant uses of Knowledge Graphs is building social graphs of users, which helps social media platforms like Facebook display users' connections with others (Chicaiza, and Díaz, 2021). Predictive Knowledge Graphs are also instrumental in enabling media platforms to offer the most relevant recommendations and content. Netflix and Facebook are examples of companies that use predictive Knowledge Graphs to suggest movies and friends to their users, respectively (Zhu, 2022).

- E-commerce: The e-commerce industry handles massive amounts of customer and product data, which require correlations to design the best marketing strategies. Knowledge Graphs have revolutionized modern e-commerce brands, and eBay's Knowledge Graph is the most popular example (Fensel *et al.*, 2020). By creating customer and product Knowledge Graphs, e-commerce businesses can leverage this technology in various ways. For instance, product Knowledge Graphs enable online stores to map the relationship between customers and products for product recommendations based on customer preferences. Brands can also analyze their product Knowledge Graphs to determine or predict the demand for specific products (Li *et al.*, 2021). Furthermore, Knowledge Graphs provide personalization opportunities by allowing businesses to dig deep into their customer behaviors and design targeted marketing strategies for a higher customer satisfaction rate.

- Cybersecurity: As the digitization of industries continues to grow, there is an increasing amount of personal information being stored and shared online, posing a significant security risk. To address this issue, industries are turning to Knowledge Graphs for secure knowledge management of their critical data. Knowledge Graphs not only enhance cybersecurity practices across various industries, but also have specific applications in the cybersecurity domain itself (Liu *et al.*, 2022). For instance, historical cyber attack data can be mapped using Knowledge Graphs to detect patterns and strengthen security practices. Additionally, cybersecurity Knowledge Graphs can be used to predict future attacks by mapping endpoints between systems to identify vulnerable areas. Furthermore, a network of cybersecurity data can help organizations holistically view their security structure to identify and fill any gaps in their security implementations, ultimately preventing malicious attacks from infiltrating their systems (Sikos, 2023). (Zou, 2020) offers a comprehensive review of Knowledge Graph applications across various domains. For a more up-to-date perspective, (Ji *et al.*, 2022) presents a survey that explores Knowledge Graphs and delves into research topics related to knowledge-aware applications.

1.3.7.2 Examples of Big Knowledge Graphs

Various companies have created different types of Knowledge Graphs for diverse purposes. While some companies use smaller, internal Knowledge Graphs for online functions, others have developed larger graphs that are widely used by people all over the world. Google, Facebook, IBM, and eBay are among the companies that have created some of the largest Knowledge Graphs to date.

- Google Knowledge Graph: The Google Knowledge Graph is one of the largest and most well-known Knowledge Graphs to date (Ehrlinger and Wöß, 2016). It was launched by Google in 2012 as a way to enhance its search engine results by providing users with more detailed and accurate information about the search query. The Google Knowledge Graph contains billions of facts and relationships between entities, such as people, places, and things, which have been gathered from a wide range of sources, including Wikipedia, Freebase (Bollacker *et al.*, 2008), and Google's own search results. The information in the Knowledge Graph is presented in a visually appealing way, with a panel that appears on the right-hand side of the search results page, providing users with a quick and easy way to access relevant information (see Figure 1.4). The Google Knowledge Graph has become an essential tool for users to get a more

comprehensive understanding of their search queries and has led to significant improvements in search results accuracy.

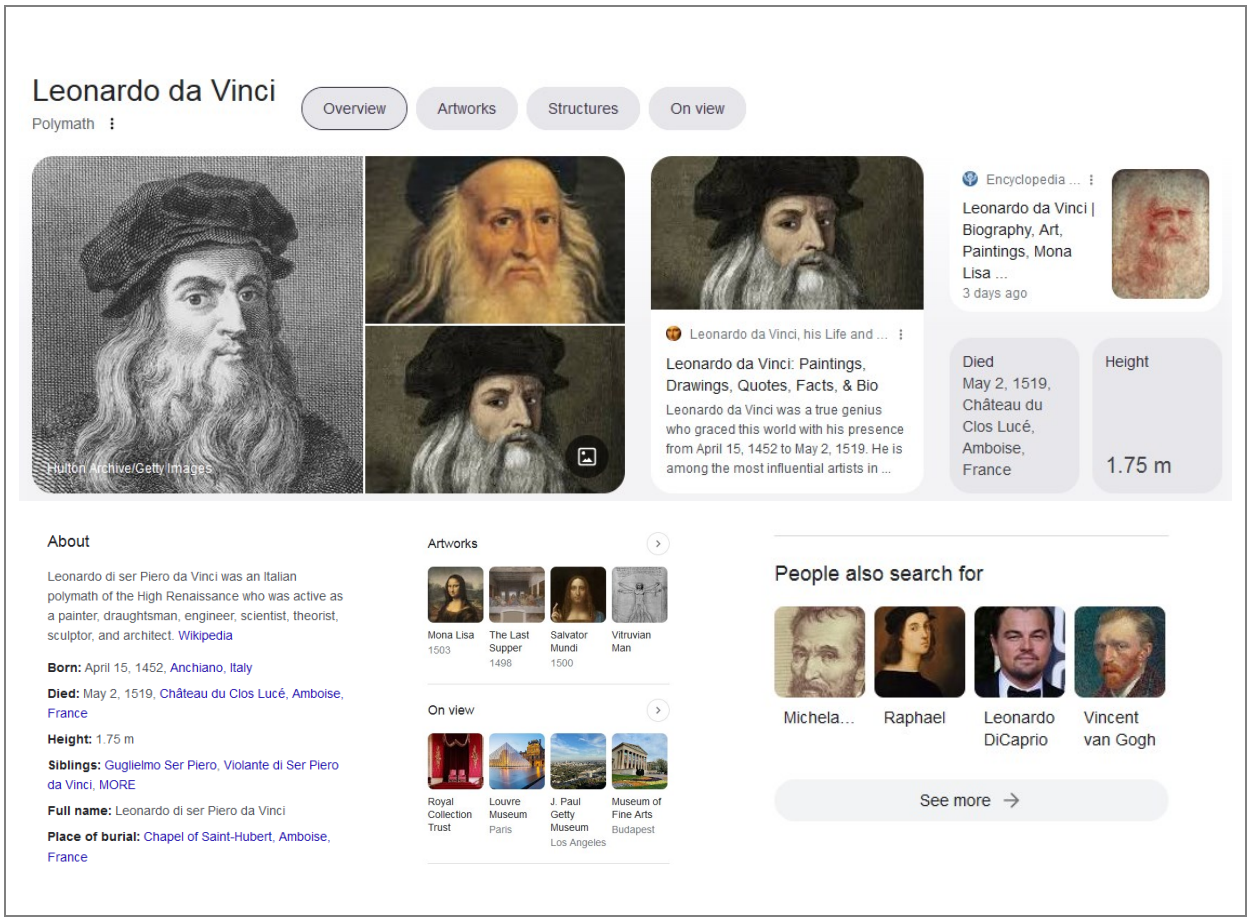


Figure 1.4. Google Knowledge Graph information for the entity Leonardo da Vinci. The panel showcases a brief textual overview of Jefferson, a link to his Wikipedia page, and a list of significant attributes that define him. Additionally, the panel displays related entities associated with Leonardo da Vinci.

- DBpedia Knowledge Graph: DBpedia (Auer *et al.*, 2007) a crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects such as Wikipedia. DBpedia allows users to query relationships and properties associated with Wikipedia resources, including links to other related resources. This structured content can be accessed and manipulated, which makes it an essential tool for machine learning, natural language processing, and other data-driven applications. DBpedia is widely used in industry and academia and is an essential resource for the Semantic Web. It is a Knowledge Graph consisting of over 5 billion facts, which are represented using RDF format, and linked to external Knowledge Graphs such as Freebase and Wikidata (Ringler and Paulheim, 2017). The data in DBpedia is available under an open license, making it accessible to anyone for use in research, commercial, or other purposes. This thesis relies on DBpedia to create a KG for

recommender systems (Chapter 3).

- Wikidata: Wikidata is a free and open knowledge base that was created by the Wikimedia Foundation in 2012. It is a collaborative project that allows anyone to contribute and edit structured data. The data in Wikidata is structured in the form of "items", which represent concepts or things in the world. Each item is assigned a unique identifier, and information about the item is organized into a set of "statements", which consist of a property-value pair. The knowledge base contains information in multiple languages, and the data is organized in such a way that it is easy to map between different languages. This makes it a valuable resource for cross-lingual applications and services (Waagmeester *et al.*, 2020). Wikidata provides a number of APIs and tools for accessing and using the data in the knowledge base. The APIs support a variety of query languages, including SPARQL, and the data is available in a variety of formats, including RDF and JSON.

- GeoNames: GeoNames is a geographical KG that contains over 25 million place names and other geographical data, such as administrative divisions, postal codes, and geological features. It is a crowdsourced project and is maintained by the GeoNames community. GeoNames is an important resource for a wide range of applications, such as geocoding, mapping, and research. It provides free access to the geographical data through various web services, including a search API, a geocoding API, and a timezone API (Zhu *et al.*, 2017) GeoNames is also used as a source of data for various Knowledge Graphs, such as DBpedia and Wikidata.

1.3.7.3 Limitations of Knowledge Graphs

Knowledge Graphs have become an increasingly popular way of representing and organizing information, particularly for use in artificial intelligence and natural language processing applications (Tiddi and Schlobach, 2022). While Knowledge Graphs can be powerful tools, they do have some limitations:

- Quality of data: The accuracy and completeness of a Knowledge Graph depends on the quality of the underlying data. If the data is incomplete or contains errors, the Knowledge Graph may not be reliable.

- Difficulty of creation: Creating a Knowledge Graph can be a time-consuming and labor-intensive process, requiring subject matter expertise and data curation skills.

- Scalability: As the size of a Knowledge Graph grows, it can become more difficult to maintain and update, and the computational resources and the computational time required to process it may increase.

1.3.7.4 Open Research Issues

Knowledge Graphs have been a topic of active research in recent years, and there are several new research issues being explored:

- Representation learning: One of the key challenges in building and using Knowledge Graphs is representing information in a way that is both expressive and computationally efficient. Representation learning techniques, such as graph embedding and Knowledge Graph completion, are being explored as a way to improve the representation of Knowledge Graph data.

- Integrating diverse data sources: Knowledge Graphs often draw on data from a wide range of sources, such as structured databases, unstructured text, and social media. Research is ongoing to develop methods for integrating these diverse sources of data and resolving conflicts and inconsistencies.

- Explainability and interpretability: As Knowledge Graphs become more widely used in applications such as natural language processing and recommendation systems, there is a growing need for methods to explain and interpret the results produced by these systems (Tiddi and Schlobach, 2022).

- Dynamic Knowledge Graphs: Many Knowledge Graphs are static, meaning that they do not change over time. However, there is growing interest in developing dynamic Knowledge Graphs that can evolve as new information becomes available (Wu *et al.*, 2022).

- Privacy and security: Knowledge Graphs often contain sensitive information, such as personal data and proprietary information. There is ongoing research into methods for protecting the privacy and security of this information while still allowing for effective use of the Knowledge Graph (Chen *et al.*, 2020).

Conclusion

In conclusion, this chapter has explored various aspects of the Semantic Web, including its definition and core technologies such as RDF, RDFS, OWL, and SPARQL. We have also discussed the importance of Linked Data and the emergence of Knowledge Graphs as a powerful tool for representing and organizing data in a more meaningful way. Furthermore, we have explored various applications of Knowledge Graphs in different domains, such as healthcare, finance, and e-commerce.

While the benefits of Knowledge Graphs are evident, we have also discussed their limitations, such as scalability, data quality, and interpretability. Despite these limitations, there are still many open research issues to be addressed in the field of Knowledge Graphs, such as how to efficiently integrate multiple data sources, how to handle dynamic and evolving data, and how to ensure privacy and security in a Knowledge Graph ecosystem.

Chapter 2

Recommender Systems Meet Knowledge Graphs

Contents

Introduction

Recommender Systems (RS) are computational algorithms that leverage user preferences and historical data to generate personalized recommendations for a wide range of items and services. This chapter explores the design and challenges associated with building effective RS. The first part discusses the definition of RS, common problems faced by RS designers, and different recommendation approaches. Additionally, it explores evaluation metrics that quantify RS performance. The second part introduces the new kind of RS known as Linked Data-based RS, which leverages the wealth of knowledge available in datasets published under Linked Data principles.

2.1 Recommender Systems

2.1.1 Recommender System Definition

Recommender Systems (RS) are filtering tools that aim to suggest relevant and personalized recommendations to the users within large sets of available options (Burke, 2002). The idea is to predict the evaluation that a user would give to some objects called items based on his past preferences. The information about these preferences is obtained through feedback provided by the user as the interaction between users and items.

In general, the fundamental building components of any recommendation system are the users, the items, and the ratings, and these are commonly organized and stored in a rating matrix denoted as *R: Users×Items* (see Table 2.1). In this matrix, each row corresponds to a particular user, and each column corresponds to a specific item. The value of each element in the matrix represents the rating that a particular user has assigned to a particular item. In the following sections, we will examine these three elements in more detail.

Table 2.1. A visual representation of *R: User x Item* matrix.

	Item 1	**Item 2**	**Item 3**	**Item 4**	**Item 5**	**Item 6**	**Item 7**
User 1	4	3	2	0	0	1	3
User 2	5	0	2	3	3	5	4
User 3	5	0	1	0	2	4	4
User 4	4	4	5	0	5	3	3

- **Items I** are a conceptual representation of the various objects or services that the platform offers, such as news articles, web pages, books, songs, and movies. Typically, these items are identified in a recommendation system using a numerical identifier *ID*, such as "*Item 1*", "*Item 2*", and so on.
- **Users U** are individuals or entities who interact with the system by providing feedback, consuming recommended items, or both.
- **Feedback R** represents the relationship between users and items. Overall, the information derived from feedback can be expressed in two ways :

***Explicit feedback*:** can be expressed in multiple ways (Schafer *et al.*, 2007):

- **Scalar ratings:** a user provides a numerical rating for a particular item, for example, when a user rates a movie on a scale of 1 to 5 stars.
- **Reviews:** a user provides textual comments describing his experience or expressing his overall opinions about a particular item. By analyzing the text of the reviews, recommender systems can extract useful information such as the user's sentiment towards the item or specific aspects of the item that the user found appealing or unappealing.
- **Binary ratings:** usually known as Like/Dislike ratings, a user indicates whether it likes or dislikes a particular item, such as a post on a social media platform. With binary ratings, users are limited to selecting from two options only, positive or negative.
- **Favorites:** a user selects certain items as their favorites, indicating a higher level of preference for those items.
- **Tags:** A user assigns tags to items, indicating their own categorization or description of the item.

Implicit feedback: refers to the user's behavior or actions that indirectly reflect their preferences or interests. Unlike explicit feedback, which is direct and intentional, implicit feedback is often collected passively by the system based on user behavior. Some common examples of implicit feedback in recommender systems include:

- **Click-through rate (CTR):** The number of times a user clicks on a recommended item.
- **View-time:** The amount of time a user spends looking at a recommended item.

- **Purchase history:** The items a user has bought or added to their cart.
- **Search history:** The terms a user searches for on the platform.
- **Watchlists:** A user adds items to their watchlist or wish list, indicating an intention to consume or purchase those items in the future. In the case of watchlists, users add items they are interested in to a list, indicating a potential desire to consume or purchase those items in the future. This action can be interpreted as a signal of user interest and used as implicit feedback.

Implicit feedback can be useful in improving the accuracy of recommender systems because it allows the system to learn from the user's behavior and adjust recommendations accordingly. However, it can also be noisy and difficult to interpret, as user behavior may not always reflect their true preferences. Therefore, it's important to consider the context and potential biases in the data when designing and training recommender systems using implicit feedback (Burke, 2000).

2.1.2 Recommendation problems

Recommendation systems are required to fulfill different criteria, and their importance depends on the specific application domain in which the RS is being used. To achieve optimal performance, various criteria must be balanced against one another. Some of these criteria are mentioned below:

- **Accuracy:** is a critical aspect of RS because it directly impacts user satisfaction and system performance. If a recommendation system fails to provide accurate recommendations, users are likely to become frustrated and stop using the system altogether. It's worth noting that achieving high accuracy in RS is not always straightforward. Factors such as data sparsity, cold-start problems, and user diversity can all make accurate prediction challenging. As a result, researchers and practitioners continue to develop and refine techniques for improving the accuracy of recommendation systems. The prediction engine at the heart of a RS calculates the usefulness of an item for a user, and a system that provides more precise predictions is likely to be favored by users. There are various metrics available to measure the accuracy of predictions, as outlined in (Shani *et al.*, 2011).
- **Coverage:** it can refer to both item coverage and user coverage. Item coverage refers to the proportion of items that the RS can recommend to users. In many cases, the majority of items

receive little attention, a phenomenon known as the long tail effect (Ge *et al.*, 2010). To overcome this, RSs may aim to promote long-tail items to improve coverage. User coverage, on the other hand, refers to the proportion of users for which the RS can provide recommendations.

- **Novelty:** it refers to the ability of the RS to recommend items that the user is not already aware of, and which have not been previously rated or seen by the user. This is particularly important in applications that require novel recommendations. Evaluating RS performance in terms of novelty can provide insights into the system's ability to generate serendipitous and engaging recommendations for users (Vargas, 2011).
- **Diversity:** it refers to the ability of the RS to recommend items that span the whole set of user preferences, and not just items related to a single user interest. In other words, recommendations should be diverse and include items that are different from each other, rather than simply recommending similar items over and over. High diversity can increase user satisfaction and engagement, leading to higher user retention and revenue for businesses. However, optimizing diversity can be challenging, as it involves balancing the need to recommend unfamiliar and diverse items with the need to recommend relevant items that match the user's preferences (Zhang and Hurley, 2008).
- **Serendipity:** it refers to the ability of the RS to recommend surprising and unexpected items that users would not have found on their own. The measure of serendipity indicates how surprising the relevant recommendations are. It is not desirable for the RS to recommend obvious or predictable options, but rather those that are unexpected and novel. Researchers have developed various metrics to measure serendipity, including those proposed by (Ge *et al.*, 2010) and (Murakami *et al.*, 2007).
- **Explainability:** it is another important criterion for recommendation systems. Indeed, users may be skeptical or mistrustful of recommendations if they do not understand how they were generated or if they suspect that the recommendations are biased or influenced by external factors. Therefore, it is important for RS to provide clear and transparent explanations for their recommendations, both to improve user trust and to ensure ethical practices. (Tintarev and Masthoff, 2015) have studied different ways to provide explanations for recommendation systems.

2.1.3 Recommender Systems Approaches

The goal is to make recommendations for a group of specific users by predicting how useful different items will be for them. Many methods have been developed to improve the accuracy of these recommendations, which can be categorized based on factors such as the type of recommendation problem being solved (e.g. rating prediction or Top-N recommendation) or the type of feedback being used (e.g. explicit or implicit feedback). However, the most commonly used classification system in this field focuses on how information about users and items is utilized to make recommendations and includes the following categories: Content-Based Filtering (CBF) approaches, Collaborative Filtering (CF) approaches, and hybrid approaches (Burke, 2000). Figure 2.1 offers a concise visual representation of the key components of each approach.

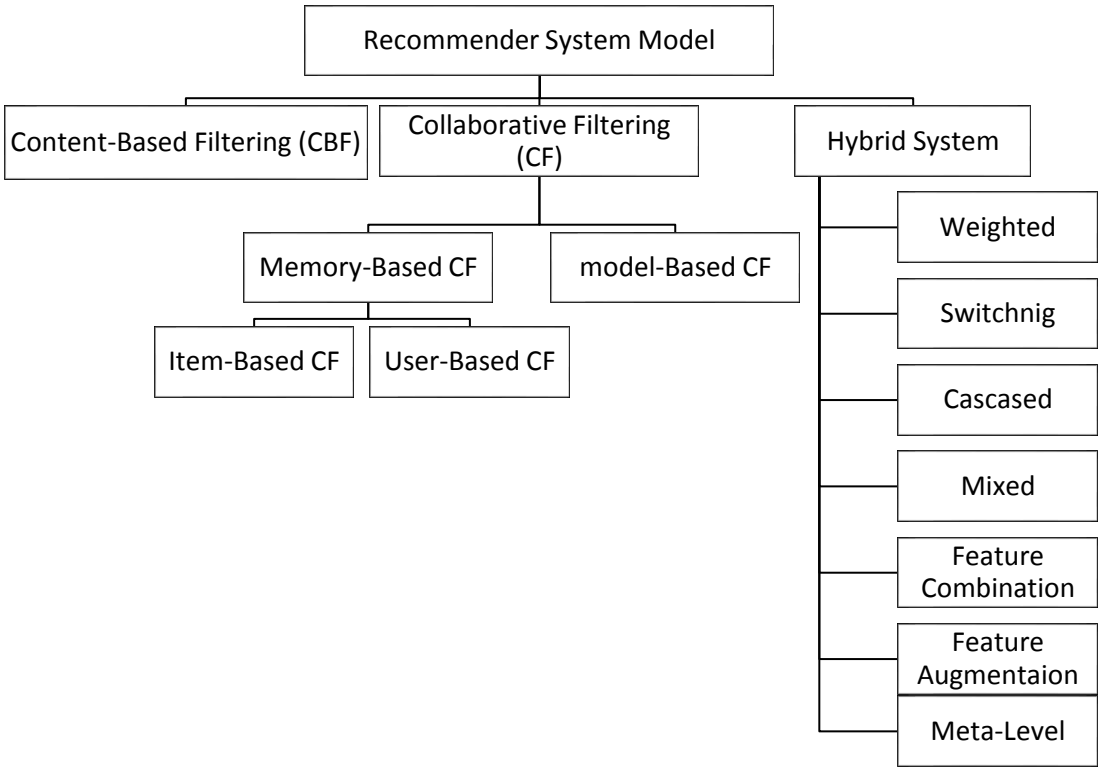


Figure 2. 1. Overview of recommendation approaches.

2.1.3.1 Content-Based Filtering Approaches

Content-Based Filtering (CBF) methods (Pazzani and Billsus, 2007) utilize information about users and items to create recommendations. This information can take various forms such as features, textual descriptions, and tags. The primary concept behind CBF is to match user and

item profiles based on the user's preferences for item attributes to suggest items that are similar to those the user previously interacted with. Implementing a CBF approach involves gathering pertinent information about the content of items, creating item and user profiles, and filtering items based on the resemblance between profiles. CBF has several advantages, such as:

- The ability to provide recommendations in item cold-start scenarios where no previous user interactions exist.
- The transparence and explainability as the recommendations are based on the content features of the items.

Despite its benefits, CBF also has some drawbacks, such as:

- The unavailability of data content, as the quality of recommendations depends on the number and type of features associated to the items. This often requires domain knowledge and enough information about the items, which may not always be available.
- The overspecialization, as it tends to propose similar items to those the user has already interacted with and is not capable of providing unexpected or serendipitous recommendations.
- The building of a consistent user profile requires sufficient user interactions, making CBF unsuitable for cases where only a few interactions are available, such as in user cold-start scenarios.

Due to these limitations, this model has found more extensive use in recommending easily recommendable items or text data items based on user profile information and item information. Various application areas where it has been predominantly employed include recommending music based on its properties (Iwahama *et al.*, 2004), movies based on their properties (Salter and Antonopoulos, 2006), e-commerce recommendations (Weihong and Yi, 2006), and recommendations for educational materials (Ghauth and Abdullah, 2010). Text mining techniques such as semantic analysis (Di Noia *et al.*, 2012a; Di Noia *et al.*, 2012b), TF-IDF (Term-Frequency Inverse Document Frequency) (Kompan and Bieliková, 2010), Neural Network (Son and Kim, 2017), Naive Bayes, and SVM (Cantador *et al.*, 2008) are utilized by the Content-Based Filtering Model to discern user preferences.

An example of a CBF approach in action is the recommendation system used by the online streaming service Netflix. Netflix uses CBF to recommend movies and TV shows to its users

based on their viewing history and other information about the content of those movies and shows.

For example, if a user frequently watches action movies, Netflix's CBF algorithm would analyze the content of those movies (e.g. the presence of car chases, gun fights, explosions, etc.) and create an "action" profile for that user. When recommending new movies or shows, the algorithm would then search for other movies or shows that have similar "action" profiles, such as other action movies starring the same actors or with similar plotlines.

Netflix's CBF algorithm also takes into account other content attributes, such as genre, director, and rating, to create a more comprehensive profile of each user's preferences. By utilizing this information, the algorithm can recommend personalized content that matches each user's individual taste.

2.1.3.2 Collaborative Filtering Approaches

The concept of Collaborative Filtering (CF) was introduced in Tapestry (Goldberg *et al.*, 1992); which is an experimental email system designed to help users filter out spam messages. The idea was that users could collaborate by annotating the usefulness of certain emails and sharing that information with others. This was the first instance of filtering information based on the feedback of other users, without any details provided on how similar users were identified or how personalization was achieved.

Following the success of CF in email filtering, numerous CF approaches were proposed for personalized recommendation. The fundamental principle behind these approaches is that users with similar interaction histories are likely to have similar preferences in the future. Recommendations are therefore based on user history and the behavior of similar users. Unlike CBF approaches, no additional information about items is needed for recommendation.

There are two main types of CF approaches: memory-based and model-based methods (Park *et al.*, 2012).

- **Memory-Based CF**

These approaches utilize the user interactions that are recorded to generate recommendations and they can be user-based or item-based, as described in (Desrosiers and Karypis, 2011). The

user-based approach recommends items to a target user by identifying other users who are most similar to the target user in terms of their past interactions. On the other hand, the item-based approach, as proposed in (Deshpande and Karypis, 2004), suggests items that are similar to the ones that the target user has previously rated or interacted with. Consider a movie recommendation system that proposes films to users based on their prior ratings as an example of a user-based approach. Two users are thought to have similar likes if they have given the same movies high ratings. They are also more likely to have similar choices in the future. Thus, when one user requests a recommendation, the system identifies the most similar users and recommends items that the similar users have enjoyed. This approach is simple and easy to implement but suffers from the sparsity problem when the number of users and items is large. While in an item-based approach, consider a movie recommendation system. The k-NN algorithm would recommend movies to a target user based on the ratings of the movies by the k users who are most similar to that target user in terms of their past movie ratings. Specifically, the algorithm would identify a set of users who have similar movie preferences to the target user, based on the similarity of their past ratings. Then, the algorithm would recommend movies that have been highly rated by those similar users, but have not yet been watched by the target user.

The similarity between users can be measured using different metrics, such as Pearson correlation coefficient (Equation 2.1), cosine similarity (Equation 2.2), or Euclidean distance (Equation 2.3). The value of k (i.e., the number of most similar users to consider) can be determined empirically or by using some optimization techniques.

- *Pearson Correlation.*

$$sim_{PC}(u,v) = \frac{\sum_{i=1}^{I_u \cap I_v}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i=1}^{I_u \cap I_v}(r_{u,i} - \bar{r}_u)^2 (r_{v,i} - \bar{r}_v)^2}} \quad (2.1)$$

Where, u, v represent two users in a dataset, i represents an item in the dataset, I_u represents the set of items that user u has rated or interacted with, I_v represents the set of items that user v has rated or interacted with, $r_{u,i}$ represents the rating or interaction of user u with item i, $\bar{r}_u$ represents the mean rating or average interaction of user u across all items in I_u, and $\bar{r}_v$ represents the mean rating or average interaction of user v across all items in I_v.

- *Cosine Smilarity.*

$$sim_{CS}(u,v) = \frac{r_u . r_v}{|r_u| . |r_v|} \quad (2.2)$$

Where, r_u and r_v represent the rating vectors of users u and v, respectively, across all items in the dataset. $|r_u|$ and $|r_v|$ represent the magnitudes or Euclidean norms of the vectors r_u and r_v, respectively.

- *Euclidean distance.*

$$sim_{ED}(u,v) = d(\vec{u},\vec{v}) = \sqrt{\sum_{k=1}^{n} (r_{u,k} - r_{v,k})^2} \quad (2.3)$$

Where, $\vec{u}$ and $\vec{v}$ represent the vector of ratings of users u and v, respectively, across all items in the dataset. $r_{u,k}$ and $r_{v,k}$ represent the rating of users u and v, respectively, with the k-th item. n represents the total number of items in the dataset.

- **Model-Based CF**

These approaches utilize the user interactions to create a predictive model, which is then used for recommendations. For instance, Netflix's movie recommendation system uses Matrix Factorization to predict the user's ratings for new movies and recommend the top-rated ones to the user. This approach is more complex but has been shown to perform better than memory-based methods in practice, especially in handling sparsity and scalability issues.

The memory-based collaborative filtering (CF) approach has both advantages and disadvantages. The advantages can be summarized in the following:

- It is a simple and easy-to-understand method for recommendation that only requires one input parameter - the number of neighbors, k.
- Recommendations are easily interpretable because they are based on similar users or items that are clearly identified.
- This approach is often used to generate statements like "Customers who bought this item also bought...".

- In comparison to user-based approaches, item-based approaches tend to be more accurate because item relationships are usually more stable than user relationships.
- Furthermore, item-based methods are more computationally efficient, especially when there are more users than available items. However, user-based approaches tend to generate more personalized recommendations and are generally more satisfying for users.

However, the disadvantages can be summarized in the following:

- Despite its simplicity, memory-based CF suffers from scalability issues and is not very sensitive to sparse data.
- For CF methods to achieve high accuracy, a sufficient amount of user interaction data must be collected and available. Both memory-based and model-based approaches require this, but model-based approaches have been found to outperform memory-based approaches in terms of accuracy.
- Therefore, model-based approaches are generally preferred in recommendation system applications.

2.1.3.3 Hybrid Approaches

CBF and CF use different sources of input to generate recommendations, and as a result, each has its own strengths and weaknesses. CBF can be overspecialized, while CF may struggle with cold-start problems. To address these limitations, hybrid methods were introduced (Burke, 2002). These methods make use of multiple sources of input simultaneously, allowing for the combination of different recommendation approaches in one framework. This approach overcomes the limitations of individual approaches used alone. There are several principles that underlie the design of these systems:

In 2002, Burke (Burke, 2002) proposed a taxonomy of seven hybridization techniques.

- **Weighted:** This technique combines multiple recommendation methods individually and distributes the recommended items evenly across all existing items in each recommendation list produced. The item with the highest score is then chosen for the final recommendation.

- **Switching:** This technique does not combine different recommendation system results, but rather dynamically selects a system based on a criterion related to the type of recommendation desired (e.g., short-term vs long-term). This criterion may be difficult to determine, so methods are also selected based on confidence criteria for the recommendation.
- **Mixed:** This technique uses multiple recommendation methods in parallel and presents a mixture of recommended items from different methods. Candidate items are selected by asking each system to provide an associated score, predicted score, and/or confidence index. A specific module then mixes these recommendations with sorting and selection based on associated candidate scores.
- **Feature Combination:** This technique uses data typically used for one type of recommendation system in another context. For example, using user review data for items typically processed by a content-based system.
- **Feature Augmentation:** This technique adds characteristic user and item data before using it as input data for the recommendation system.
- **Cascade:** The cascade hybridization method is hierarchical, where each recommendation method refines a recommendation obtained by a previous recommendation method.
- **Meta-Level:** This method takes as input the model made by another recommendation system. Compared to feature augmentation, this approach requires the complete replacement of the input model with the output of the previous recommendation.

2.1.4 Evaluation Metrics of Recommendation Systems

In order to evaluate the performance of the Recommendation System model, it is necessary to quantify the characteristics of an excellent recommendation system (Fayyaz *et al.*, 2020). Various metrics have been developed such as: RMSE (Root Mean Squared Error), MAE (Mean Absolute Error), Precision, Recall, Accuracy, F-Measure, Receiver Operating Characteristic (ROC) Curve, and AUC (Area Under the Curve).

- RMSE. is an index commonly used to evaluate prediction accuracy and is obtained by taking the square root of the MSE calculated by dividing the sum of the squares of the difference between the actual grade and the predicted grade by the total number of grades that have been predicted (Bag *et al.*, 2019) RMSE is computed using Equation 2.4.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(actual\ value\ i - predicted\ value\ i)^2} \quad (2.4)$$

where:

- n is the number of observations or data points;
- $actual\ value\ i$ is the i[th] actual value in the dataset;
- $predicated\ value\ i$ is the i[th] predicted value generated by the model or algorithm.

- MAE measures the accuracy by calculating the average of the absolute differences between the predicted values and the actual values, it is given by Equation 2.5.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}(|actual\ value\ i - predicted\ value\ j|)^2 \quad (2.5)$$

- Precision: it measures the proportion of true positive predictions among all positive predictions made by the model. It represents how accurate the positive predictions are. The precision is given by Equation 2.6.

$$Precision = \frac{true\ positives}{true\ positives + false\ positives} \quad (2.6)$$

- Recall: it measures the proportion of true positive predictions among all actual positive instances in the data. It represents how well the model can identify positive instances. It can be obtained by Equation 2.7.

$$Recall = \frac{true\ positives}{true\ positives + false\ nagatives} \quad (2.7)$$

- F1 score: it is a weighted harmonic mean of precision and recall that balances both metrics. The F1 score combines precision and recall into a single score that provides a more complete picture of the model's performance. It can be computed using equation (2.8).

$$F1\ score = 2\frac{precision * recall}{precision + recall} \quad (2.8)$$

The F1 score ranges from 0 (worst) to 1 (best) and is a useful metric when both precision and recall are important.

- ROC (Receiver Operating Characteristic) curve: it is a graphical plot that illustrates the performance of a binary classification model at different classification thresholds. It demonstrates the balance between the true positive rate (sensitivity) and false positive rate for all possible threshold values. An ideal model would have a ROC curve that goes through the top-left corner of the plot with a true positive rate of 1 and a false positive rate of 0.

- AUC (Area Under the Curve) is a metric that measures the overall performance of a binary classification model using the ROC curve. AUC represents the area under the ROC curve and ranges from 0 to 1, with higher values indicating better performance. An AUC value of 0.5 indicates that the model performs no better than random guessing, while an AUC value of 1 indicates perfect classification performance.

2.2 Feeding Recommender Systems with Linked Data

2.2.1 Semantic-aware Recommender Systems

The Semantic Web provides a standardized way to represent domain knowledge, which can be used to enhance RSs. One way the Semantic Web can enhance RSs is by providing a more detailed and structured representation of items, users, and their preferences (Di Noia *et al.*, 2016). In traditional RSs, items are often represented simply as a set of attributes or features, and users are represented by their ratings or behavior. However, in the Semantic Web, items and users can be represented as entities with rich descriptions and relationships to other entities.

For example, an item in an e-commerce site might be represented as a product entity with properties such as name, brand, category, and price. The product entity might also be linked to other entities such as a manufacturer, a distributor, or a related product. Similarly, a user might be represented as a customer entity with properties such as name, age, location, and purchase history. The customer entity might also be linked to other entities such as a social network profile or a loyalty program.

Another way, the Semantic Web can improve RSs is by providing a more flexible and adaptable framework for modeling and analyzing data. Traditional RSs often use rigid algorithms such as collaborative filtering or content-based filtering, which can be limited by

their assumptions and biases. However, in the Semantic Web, data can be modeled and analyzed using a variety of techniques such as Graph Analysis, Machine Learning, and Natural Language Processing. For example, a RS might use Graph Analysis to identify clusters of items or users based on their relationships in a KG. It might use Machine Learning to train a model that can predict a user's preferences based on their behavior and context.

In the second part of this chapter, our focus shifts to presenting a comprehensive categorization of the state-of-the-art semantic aware recommender systems. We aim to organize the existing works in this field into distinct categories, namely ontology-based RS and knowledge-aware RS. Moreover, within the knowledge-aware RS category, we further subdivide the works based on the employed embedding technique. These techniques encompass Graph Embedding Methods, Knowledge Embedding Methods, and deep graph embedding methods. Semantic-aware RSs' hierarchy is shown in Figure 2.2. The numerous elements and subcategories that make up the recommendation domain are visually summarized by this hierarchical depiction.

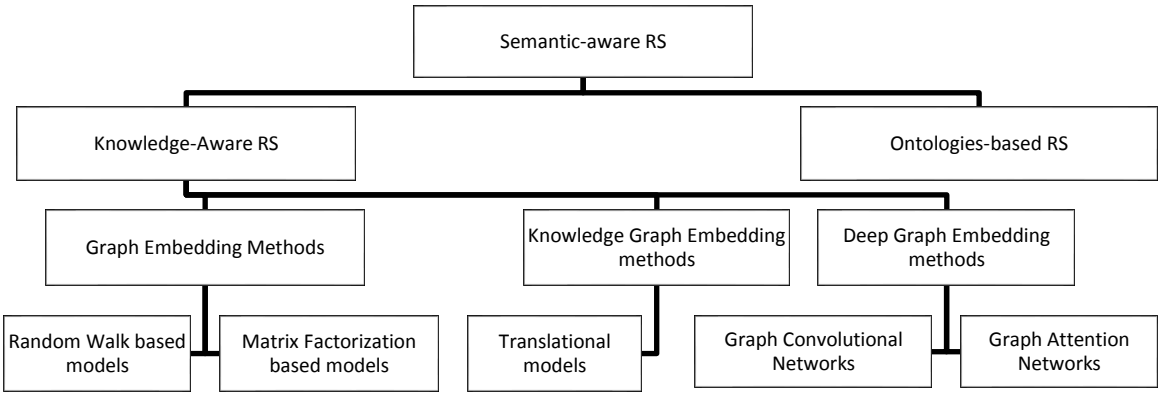


Figure 2.2. Semantic-aware Recommender Systems hierarchy.

2.2.1.1 Ontology-Based Recommender Systems

An ontology is a conceptualization of a domain into a human-understandable and machine-readable format. It is made up of entities, attributes, relationships, and axioms that describe a domain (Middleton, *et al.*, 2009). Within an organization, an ontology can be used to provide a comprehensive understanding of a domain and its related activities.

Ontology-based RSs are a specific category of RSs that utilize ontologies as a means to model and represent knowledge pertaining to items and their interrelationships. The

fundamental concept behind ontology-based recommendation is to establish a shared ontology, allowing for the representation of both recommended items and user preferences. This is achieved by transforming item descriptions, which are typically keyword-based, into descriptions that are concept-based.

Through this transformation process, the system is able to generate more accurate and relevant recommendations. For instance, consider an ontology that incorporates information regarding a user's preferences for a particular music genre. Through the utilization of the ontology, the system gains the ability to tap into the user's preference data and make recommendations that are finely tuned to their specific music genre of interest. This enables the system to suggest highly relevant songs or albums within that genre, enhancing the overall recommendation experience for the user.

Ontology-based recommendation has been applied in various domains, including e-commerce, healthcare, education, and many others. While each domain has its own unique challenges and opportunities, e-commerce is one of the most widely researched and well-established domains for ontology-based RSs. In this section, we will present notable state-of-the-art works in the field of ontology-based recommendation. These works have been applied in various domains, including e-commerce, healthcare, education, and many others. For example, (Guia *et al.*, 2019) presented an e-commerce RS, which combines CF with ontology-based recommender; the proposed RS takes into account not only the preferences of users similar to the active user but also leverages ontological information about the user, their neighbors, and the interconnections between them. Another ontology-based RS is proposed in (Ayundhita *et al.*, 2019). It aims to recommend laptops to users who are not aware of low-level specifications. The proposed approach uses the ontology to map functional requirements with low-level specifications. Basically, the system asks functional requirements to the user according to their preferences and then the ontology maps those requirements with low-level specifications to find laptops that match the active user’s preferences. Although it is an innovative idea, this approach required an effort of the active user, who needs to introduce the functional requirements of the intended laptop.

In (Karthik and Ganapathy, 2021), the authors proposed a new fuzzy logic-based product RS. Specifically, the paper introduced an algorithm for calculating the sentimental score of a product based on its target category and end user. To enhance the accuracy of the system, the authors suggested using a fuzzy rule and ontology-based approach, which incorporates

ontology alignment to enable dynamic prediction based on the search context. Baizal *et al.* (Baizal *et al.*, 2016) expanded the utilization of ontology-based recommendation techniques by incorporating product reviews alongside product features. The authors put forward the idea of expressing customer preferences as functional requirements for the product. These requirements are then mapped to corresponding product features and reviews, which are stored in an ontology. Through exploration of the semantic relationships within this ontology, the proposed recommendation system can offer suggestions that align with the specific needs of the customers. Recently, (Yin *et al.*, 2023) suggested a novel approach in fresh e-commerce logistics, wherein an ontology-based package design system is implemented to tailor the packaging of perishable items. The package design can be automatically recommended, taking into account the unique contextual parameters of each order to ensure that all package design prerequisites are met.

Another domain of application for ontology-based RS is e-learning. In this domain, the ontology can be used to model learners and learning resources like courses, textbooks, instructional materials, and student characteristics such as their previous coursework, learning styles and academic goals. For example, (Ouf *et al.*, 2017) developed a framework for personalizing e-learning resources by proposing four ontologies to represent the learner, learner object, learning activities, and teaching method. Furthermore, the personalization dimension is added by using rules over the ontologies. In (Obeid *et al.*, 2017), the authors suggested a hybrid RS that suggest universities to the students using an ontology to model domain knowledge regarding universities and students. The proposed method is unique in that it goes beyond simply assessing students based on their grades and takes into consideration their skills and interests as well. A similar work was provided in (Tarus *et al.*, 2018a) that presented a hybrid method of providing educational recommendation by implementing an ontology over the learners and learning resources. Sequential pattern mining is utilized to detect the learner's past sequential behavior from weblogs. The authors of (Zehra *et al.*, 2017) suggested a method for matching schools to the preferences of an active user through an ontology-based recommender system. This approach leveraged the active user's comments on their Facebook feed related to a particular school to extract features, which are used to develop the ontology-based recommender system based on the polarity of the comments.

A detailed review of several ontology-based RSs is presented in (Tarus *et al.*, 2018b) during the period 2005 to 2014. The main discovery of the authors is that the majority of ontology-

based RSs reviewed employ domain ontology and web ontology language as their modeling approach. Another review is presented in (George and Lal, 2019), researchers provide an overview of the research from 2010 to 2018 using ontology to achieve personalization in RSs in the e-learning domain. They found that using ontology-based RSs for e-learning services can yield better results than conventional recommendation techniques, this approach also has certain drawbacks, such as being time-consuming and requiring specialized knowledge engineering skills. They further suggested that hybridizing traditional techniques with ontology-based recommenders can be more effective than any other method in the e-learning domain.

(Rahayu *et al.*, 2022) presented a more recent analysis of ontology-based recommenders' research papers for personalized learning from 2010 to 2020. The review is directed to explore the development of ontology-based RSs in personalized learning. The authors' main finding is that the learning object is currently the most widely recommended item, and it can be customized using either internal or external data. In the years ahead, the learning path and feedback are expected to become increasingly popular as recommended items for students.

Healthcare ontology-based recommendation is another example of the potential benefits of ontologies. For instance, (Mckensy-Sambola *et al.*, 2022) developed a recommendation engine that can detect varying degrees of overweight and obesity in users and suggest dietary strategies to address these issues. To achieve this, the authors created an Ontology of Dietary Recommendations (ODR) with specific axioms for modeling ingredients, recipes, and a range of diets aimed at supporting individuals with obesity. (Nassabi *et al.*, 2014) proposed an ontology-based RS to promote physical activity for pre-frail elderly. In this work, an ontology for physical activities is created. Alongside this ontology, the authors define a series of rules that can be applied to the ontology to extract relevant information. Integrating the ontology and rules enabled to establish a comprehensive knowledge base that can be utilized to provide tailored exercise recommendations.

While ontology-based RSs have many potential benefits, there are also some limitations that need to be considered. One major limitation is the rigidity that defines ontologies since they usually cover a specific domain. Additionally, building an ontology-based recommender can be quite cumbersome and costly process, which requires a high degree of expertise in Knowledge Engineering. Finally, maintaining an ontology-based RS can be costly and time-

consuming. The ontology must be kept up-to-date with the latest information and changes in the domain, which can require significant resources.

2.2.1.2 Knowledge-Aware Recommender Systems

Thanks to the Open Linked Data initiative, more and more data has been published on the web using semantic web technologies, enabling the born of a new recommendation research field. In contrast to ontology-based recommendation which focuses on a specific domain, Linked Data-based recommendation can cover a wider range of domains and data sources.

The first technical tracks on Linked Data RSs have been primarily focused on leveraging LD and ML algorithms used in Information Retrieval (IR) research field. For example, (Di Noia *et al.*, 2012a) extended the well-known Vector Space Model (VSM) (Salton *et al.*, 1975) widely used in the IR domain for modeling the items within a KG. In this work, the VSM approach represents a textual document d as a vector with n dimensions, where, each element in the vector represents a specific TF-IDF (Term-Frequency Inverse Document Frequency) weight. Another interesting direction about the usage of LOD for content-based RSs is explored in (Musto *et al.*, 2013) where the authors presented Contextual VSM, a content-based context-aware recommendation framework that adopts a semantic representation based on distributional models and entity linking techniques.

Mirizzi *et al.* introduced MORE (Mirizzi *et al.*, 2012), a Facebook application for movie recommendation. MORE employs an RS that harnesses the power of two KGs, DBpedia and LinkedMDB, to deliver personalized movie suggestions. Upon installation, users can effortlessly search for movies by typing a few initial characters, and the system promptly provides a curated list of matching movies sourced from DBpedia. To generate a refined list of recommendations, MORE collects each user's favorite movies and applies PageRank (Page *et al.*, 1999) and K-NN algorithms. The authors adopted a three-dimensional matrix representation of the semantic properties, where each vector in the matrix represents a property. With this representation and using a semantic version of SVM classifier, the authors were able to measure the similarity between two movies by calculating the cosine angle between the corresponding property vectors.

Similarly, in (Di Noia *et al.*, 2012b), the authors utilized the matrix representation of properties and the semantic version of the SVM classifier proposed in (Mirizzi *et al.*, 2012) to

construct a LOD-based movie RS. In this approach, a new similarity calculation function was introduced, it assigns weights to each semantic property based on its importance in the user's profile. The weights are adjusted using a genetic algorithm.

Even though various research works related to Linked Data-based RS have been published, the main similarity is that they involve creating a Knowledge Graph by matching the collaborative information encoded in the pointed-out datasets (user-item information) and the knowledge encoded in LOD Cloud (semantic meta-data), hence the new name of this category of research field, Knowledge Graph-based Recommender Systems.

The major difficulty with early approaches is that classic ML methods may struggle to effectively handle the complex patterns within KGs, particularly in terms of scalability. KGs can be massive, containing millions or even billions of nodes and edges, which poses challenges for traditional ML algorithms in terms of efficient processing and inference. This limitation impairs the effectiveness of classic ML methods and consequently results in decreased accuracy in KG-based RS. To overcome these challenges and improve RS accuracy, it is crucial to employ more effective methods able to handle KGs and their intricate patterns, enabling scalable and accurate recommendations.

Graph Representation Learning (GRL) approaches, also known as Embedding approaches, are a class of ML algorithms that have emerged as powerful techniques enabling the representation of graph-structured data in a continuous vector space. These representations can be used to solve a variety of downstream tasks, such as node classification, link prediction, and community detection.

One of the key advantages of GRL methods is their ability to address the challenge of scalability. As mentioned earlier, classic ML algorithms often face difficulties with the computational demands imposed by large-scale graphs, thereby limiting their applicability in real-world scenarios. However, GRL methods have introduced scalable solutions based on neural network architectures and optimization techniques that can effectively handle big graphs.

Researchers have considered using GRL methods to learn low-dimensional vectors for nodes and/or edges in KGs. They have focused initially on using existing graph learning methods and adapting them to KGs for recommendation purposes or on directly applying Knowledge Graph Embedding (KGE) methods specifically designed to handle KGs.

In the context of RSs, GRL methods have proven particularly valuable. By learning representations that encode the structural properties of the user-item graph, GRL methods enable more accurate and personalized recommendations, enhancing the user experience and improving the quality of recommendations. However, it is important to note that while GRL methods employ nonlinear mapping functions, they might not be explicitly designed to analyze semantic graphs. Semantic graphs are characterized by highly meaningful information and often contain rich, structured knowledge that goes beyond basic node and edges. Consequently, the complexity and depth of semantic graphs may present challenges for GRL methods in capturing their nuanced semantics and extracting comprehensive insights.

One promising approach to address this challenge is the use of Deep Graph Representation Learning (DGRL), particularly Graph Neural Networks (GNNs). Although not explicitly designed for KGs, these methods have demonstrated great promise in learning from complex graph structures. Their deep neural architectures and techniques enable the extraction of meaningful representations from intricate data, capturing complex patterns and relationships. In light of this success, the emergence of KG-based RS has become inevitable.

The following subsections outline the three primary categories of state-of-the-art KG-aware recommendation research, categorized based on different Graph Embedding methods.

2.2.1.2.1 Graph Embedding Methods

Graph Embedding (GE) methods aim to learn low-dimensional vector representations -also called embeddings- from the network structure, such that nodes that are close in the embedding space are also likely to be similar or connected in the original graph.

One popular branch of GE methods is based on random walk technique, where random walks are performed on the graph and the resulting node sequences are used to learn the embeddings such as:

- **DeepWalk** (Perozzi *et al.*, 2014): DeepWalk is an unsupervised learning method that learns latent node representations in a graph by treating random walks as "sentences" and using Natural Language Processing techniques to create distributed representations of nodes. The approach generates random walks from each node and utilizes a skip-gram model to learn

node embeddings by predicting the context nodes. These embeddings are learned as the parameters of the skip-gram model.

- **Node2vec** (Grover and Leskovec, 2016): Node2vec is a widely used algorithm that learns node embeddings in a graph by using a biased random walk approach similar to DeepWalk. However, Node2vec introduces an additional parameter to balance between exploring new areas and revisiting old ones. This biased random walk explores both local and global neighborhoods using breadth-first and depth-first search strategies. After generating random walks, Node2vec trains a skip-gram model to predict the probability of observing a node's context given its embedding. The optimization objective is to maximize the likelihood of observing context nodes for each target node across all random walk sequences. Node2vec is highly scalable and flexible, accommodating unweighted and weighted graphs with different types of node and edge attributes.

The other branch of network embedding methods is based on matrix factorization, such as:

- **Graph-Regularized Embedding Propagation** (**GraRep)** (Cao *et al.*, 2015): GraRep is a GE method that represents a graph as a matrix of co-occurrence counts between nodes. It applies Matrix Factorization and transformation steps to this co-occurrence matrix to obtain low-dimensional embeddings for the nodes. GraRep uses higher-order co-occurrence information to capture the structural properties of the graph by computing higher-order co-occurrence matrices. It applies SVD to each of these matrices to obtain low-dimensional embeddings for the nodes, which are concatenated to form the final representation for each node. GraRep can handle both directed and undirected graphs, as well as graphs with weighted edges.

- **Graph Factorization** (GF) (Ahmed *et al.*, 2013): GF is a method for GE that factorizes the adjacency matrix of a graph into two low-rank matrices representing the embeddings of the row and column nodes. The embeddings are obtained by minimizing a loss function that measures the reconstruction error between the original adjacency matrix and its factorization, using gradient descent to iteratively update the embeddings of the row and column nodes. GF can handle directed and weighted graphs and can incorporate additional information such as node attributes and edge features.

- **Laplacian Eigenmap** (Belkin and Niyogi, 2002): This GE method works by constructing a graph from the input data, computing the Laplacian matrix of the graph, and diagonalizing the matrix to obtain its eigenvectors, which are used as the embedding coordinates in the low-dimensional space. The method has several advantages over other dimensionality reduction techniques such as Principal Component Analysis (PCA) and Multi-Dimensional Scaling (MDS).

In the context of recommendation, it is noteworthy to mention that Node2vec has emerged as the most prominent GE method utilized in RSs. For example, HERec (Shi *et al.*, 2019) is a RS that learns user and item embeddings using different meta-paths and fuses them for better performance. It first finds the co-occurrence of users and items using meta-path guided random walks on a user-item heterogeneous graph. Then, it uses Node2vec to learn preliminary embeddings from the co-occurrence sequences. Since embeddings under different meta-paths contain different semantic information, HERec designs a fusion function to unify multiple embeddings for better recommendation performance. Finally, a prediction layer is used to predict the items that users prefer. In their work, Ragab and El-Kafrawy (Ragab and El-Kafrawy, 2022) presented a content-based recommendation system (RS) for movie recommendations. Their approach utilizes Node2vec for generating item embeddings.

Chen et al., (Chen *et al.*, 2019) introduced a new clustering recommender system called N2VSCDNNR, which is based on the Node2vec algorithm and was found to have great potential in addressing sparsity and cold-start issues. In Entity2rec (Palumbo *et al.*, 2017), the authors applied Node2vec to learn property-specific vector representations of nodes, one property at a time. They defined a sub-graph for each property as the set of entities connected by that property, to capture different semantic values of properties. Cai et al., (Cai *et al.*, 2018) also utilized the Node2vec to learn vector representations from a bibliographic knowledge graph for personalized citation recommendation.

In summary, while GRL methods have been effective in learning representations that capture the structural properties of a user-item graph, resulting in more accurate and personalized recommendations that enhance the user experience and improve recommendation quality, their linearity can limit their ability to capture complex patterns in the data, which can result in the loss of important semantic information.

2.2.1.2.2 Knowledge Graph Embedding Methods

Knowledge Graph Embedding (KGE) methods are distinct from other categories such as GRL and Deep GRL, as they designed specifically to learn vector representations of entities and relationships in semantic graphs. Translation models belong to the category of KGE methods, which use low-dimensional vectors to represent entities and relations in a KG. In these models, the interaction between entities and relations is defined as a translation in the vector space. The most commonly used translation model is TransE, which was introduced by Bordes et al. (Bordes *et al.*, 2013). In TransE, each entity and relation is represented as a vector in the same low-dimensional space. The interaction between an entity and a relation is defined as the translation of the entity vector by the relation vector. Mathematically, the score of a triple (h, r, t), where h is the head entity, r is the relation, and t is the tail entity, is defined as (see Equation 2.9):

$$score(h, r, t) = \|h + r - t\| \tag{2.9}$$

where $\|.\|$ denotes the L1 or L2 norm of the vector. The model is trained to minimize the score of the correct triples and maximize the score of the incorrect ones.

TransH (Wang *et al.*, 2014) and TransR (Lin *et al.*, 2015a) are extensions of TransE that aim to address some of its limitations. TransH introduces a hyperplane to represent each relation, and the translation is performed on the hyperplane instead of the entity vector. This allows the model to better capture the heterogeneity of relations. TransR takes this idea further by projecting the entities and relations into different spaces, and performing the translation in the relation space.

Translational models have found application in RS. For example, Wang et al. (Wang *et al.*, 2018b) introduced a news RS called DKN, which models news by combining textual embeddings of sentences learned with Convolutional Neural Networks (CNN) (Kim, 2014), and knowledge-level embeddings of entities in news content via TransD. In Zhang et al. (Zhang *et al.*, 2016), CKE is proposed as a RS that incorporates various types of side information into the CF framework, including item attribute-level feature, textual feature, and visual feature. The attribute-level feature is encoded using TransR to learn structural knowledge from knowledge graphs, while the textual and visual features are extracted using an auto-encoder. The objective function of these three feature learning modules is incorporated into the recommendation

module, allowing for joint learning of the parameters. Zhang et al. (Zhang *et al.*, 2018) introduced CFKG, a RS that builds a user-item KG. CFKG uses the TransE model to encode and learn embeddings for both relations and entities on the graph. Similarly, Cao et al. (Cao *et al.*, 2019) employed TransH to learn entity and relation embeddings for their RS called KTUP.

A systematic review of existing KGE techniques is presented in (Wang *et al.*, 2017). The authors showcased examples of three applications: relation extraction, question answering, and Recommender Systems, to illustrate how KGE can be utilized in these areas. However, the review did not offer a systematic evaluation of these tasks or present the latest advancements. Rather, the focus was on demonstrating the potential of KGE in these domains. (Guo *et al.*, 2022) presented a more recent analysis of KGE-based RSs. The survey demonstrated how various methods use KGs as supplementary information to enhance recommendation outcomes and provide interpretability during the recommendation process. Furthermore, it presented an overview of datasets utilized in different recommendation scenarios.

As a summary, KGE methods remain an effective class of KGE models that have marked recommendation research field. However, they still have some limitations. For example, they assume that the interaction between entities and relations is linear and additive, which may not always be the case. Also, they do not explicitly model the type constraints between entities and relations. Despite attempts to solve these problems, the proposed solutions came with additional computational cost and complexity.

2.2.1.2.3 Deep Graph Representation Learning Methods

In the context of KG-based RS, to make sense of this complex data and provide meaningful recommendations, it becomes necessary to employ more effective methods. These methods should be compatible with the new levels of complexity and scalability of KGs and the intricate patterns they contain.

One promising approach to address this challenge is the use of Deep Graph Representation Learning (DGRL), especially Graph Neural Networks (GNNs). While not specifically designed for KGs, these methods have shown great promise in learning from complex graph structures. Their deep neural architectures and techniques enable the extraction of meaningful representations from complex data, capturing intricate patterns and relationships. The basic idea behind GNNs is to learn representations for each node in the graph by aggregating information

from its neighbors. This process is repeated for multiple layers, with each layer updating the node representations based on the previous layer's output. By doing so, GNNs can capture both local and global structural information about the graph. There are mainly two classes of GNNs architectures:

- Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017)**:** GCNs are a type of neural network that operates on graph-structured data. It is inspired by Convolutional Neural Networks (CNNs) (LeCun and Bengio, 1998), Recurrent Neural Networks (RNNs) (Hochreiter and Schmidhuber, 1997), and auto-encoders (Vincent *et al.*, 2010). GCNs are based on the convolution operation, which is performed using the graph Laplacian matrix to represent the graph structure. In GCNs, information is propagated from a given node to its adjacent nodes through message passing, and subsequently, the node's representation is updated by computing a weighted sum of the received messages. Non-linear activation functions, such as Rectified Linear Unit (ReLU) or Tanh, are used to introduce non-linearity into the model.

The application of GCNs in RS is a notable example. For example, *Ying et al.* (Ying *et al.*, 2018) developed a data-efficient large-scale GCN engine called PinSage. The algorithm combines GCNs and Radom Walks to generate item embeddings. The Random Walk sampling technique is used to measure the importance of each node's neighbors in such a way that the Top-N nodes with the highest L1-normalized visited counts will be chosen. PinSage applies several convolutional modules to aggregate the visual and textual features from the node's local graph neighborhood to generate the embedding of an item. However, the model does not encode semantic information from the edges of the KG.

RippleNet (Wang *et al.*, 2018a) is the first attempt to apply deep propagation on a knowledge graph for recommendation purposes. RippleNet incorporates KGs into collaborative filtering to enrich the items' representation to alleviate the sparsity and cold-starting issues that collaborative recommenders suffer from. The model iteratively propagates user preferences over the KG to learn the user's multi-hop interests. Although RippleNet attempts to characterize the relationships between the user and the KG entities, the importance of the relation is not well captured.

The KGCN framework proposed by Wang et al. (Wang *et al.*, 2019a) is an end-to-end RS that automatically discovers the high-order structure information and semantic information from the KG. The key idea of KGCN is to aggregate and incorporate neighborhood information with a bias when computing the representation of a given node in the KG. It is implemented by weighting the neighbors by scores that depend on the connection relations and semantic information.

Extending the previously mentioned work, Wang *et al.* (Wang *et al.*, 2019b) proposed KGCN-LS, which uses the label smoothing assumption to rely on edge weight regularization. The Dual Knowledge Multimodal Network (DKMN) (Ouyang *et al.*, 2022) is an extension of KGNN-LS that aims to incorporate KGs and other multimodal data such as textual reviews and pictures for recommendation purposes.

In social e-commerce recommendation, Xu *et al.* (Xu *et al.*, 2019) extended the paradigm of GCN-based recommender systems to heterogeneous graphs with relation-aware aggregators and co-attentive embedding fusion mechanisms. The proposed RecoGCN model significantly improves the performance of social e-commerce recommendations. The authors in (Ma *et al.*, 2022) proposed a recommendation method named KR-GCN, which aims to tackle the error propagation problem. The authors designed a transition-based method to determine the triple scores and used a path-level self-attention mechanism to discriminate the contributions of different selected paths, which improves the relevance of the final explanation. Recently, an intelligent background music system based on DL and Internet of Things (IoT) technology is presented in (Wen, 2021). They proposed a feature extraction algorithm based on the mid-level feature structure that extracts the underlying features from the scene images. Subsequently, the critical functional components of the intelligent background music system are explained. A comprehensive survey of all major deep learning-based Point-of-Interest (PoI) recommendations is presented in (Ashraful Islam *et al.*, 2022). This work categorizes and

critically analyzes recent PoI recommendation work based on different DL paradigms and other relevant features.

- **Graph Attention Networks (GATs)** (Veličković *et al.*, 2017): GATs are another type of Neural Network that operates on graph-structured data. GATs also use message passing between nodes, but unlike GCNs, GATs use attention mechanisms to weight the messages. Attention mechanisms enable GATs to assign different importance weights to each neighbor node based on their relevance to the target node. This makes GATs more expressive and better suited for capturing complex relationships between nodes in a graph.

GATs have been used for recommendation, as evidenced by its successful application in several notable works. KGAT (Wang *et al.*, 2019) is the first system to extend GATs to knowledge graphs for recommendation purposes. Based on a self-attention mechanism, KGAT provides explainable recommendations considering high-order relationships. In the KGAT model, the explanations can be obtained directly from the model by interpreting the attention weights. In (Shimizu *et al.*, 2022), a new approach to explainable recommendation is presented, leveraging an advanced knowledge graph attention network model that takes into account item-specific side information to deliver highly accurate recommendations. The proposed framework enables direct interpretation of the reasoning behind each recommendation by visualizing the factors that contributed to it. (Dai *et al.*, 2022) proposed COAT, a novel framework with collaborative and attentive graph convolutional networks for personalized knowledge-aware recommendation. Particularly, we model the user-item graph and the KG separately and simultaneously with an efficient GCN and a personalized knowledge graph attention network, where the former aims to extract informative collaborative signals, while the latter is designed to capture fine-grained semantics. (Tu *et al.*, 2021) proposed KCAN Knowledge-aware Conditional Attention Networks, which is an end-to-end model to incorporate knowledge graph

into a RS. The authors, uses a knowledge-aware attention propagation manner to acquire the node representation Then knowledge-aware attentions are used to extract the knowledge graph into the target-specific subgraph.

The Contextualized Graph Attention Network (CGAT), introduced by Liu et al. (Liu *et al.*, 2021), leverages both local and non-local graph context information of knowledge graph (KG) entities. To capture the local context (one-hop), a user-specific graph attention technique is employed, which aggregates the relation-aware neighborhood information of an item. To incorporate the non-local context, essential entities for the target entity are extracted from the entire KG using a biased random walk approach.

The Knowledge Graph-based Intent Network (KGIN) (Wang *et al.*, 2021) focuses on understanding user intents underlying user-item interactions. It achieves this by leveraging the item knowledge graph (KG) to enhance recommendation performance and make the recommendations more interpretable. This method represents each intent as a combination of KG relations and introduces a novel relational path-aware aggregation scheme. This scheme effectively aggregates long-range connectivity relation sequences and refines multi-hop path representations to capture the intricate nature of user intents. HAGERec (Yang and Dong, 2020), is a hierarchical attention graph convolutional network to aggregate more hops. The system uses an attention mechanism to select relevant neighbor entities, while explanations are endowed to the model by building knowledge-aware connectivity. Recently, (Jin *et al.*, 2023), introduced a Meta-path guided Graph Attention Network to provide the explainable medical herb recommendations.

Deep Graph Representation Learning offers an improved method for handling complex patterns present in scalable heterogeneous semantic data. However, there are still notable challenges that must be addressed to fully unlock the potential of such data. One particular challenge is the semantic loss problem, which has not received sufficient attention. This issue

results in a lack of interpretability, making it challenging to understand how embeddings are learned and their contribution to the recommendation process. The problem arises from the non-linear transformations that occur across layers, which play a crucial role in extracting intricate patterns from this type of data.

To assist readers in exploring the relevant literature, we have summarized and organized these papers in Table 2.2 presenting an overview of the existing state-of-the-art work in KG-based recommendation. The table encompasses details such as the embedding methods employed, the specific issues tackled in each model, the datasets utilized, the recommendation scenarios considered, and the external KGs incorporated. Notably, the models are arranged chronologically, from the earliest to the most recent.

Table 2.2. A comprehensive overview of the current state-of-the-art research in KG-based recommendation. The table presents key information, including the employed embedding methods, specific issues addressed in each model, utilized datasets, considered recommendation scenarios, and incorporated external KGs. The models are arranged chronologically, from the earliest to the most recent, providing a timeline of advancements in the field. It should be noted that the presence of empty cells in the "External KG" column indicates that the respective model has generated its own KG rather than relying on an external knowledge graph.

Model	Category	KG Embed.	Solved Problem	Dataset	External KG	Scenario
CKE (Zhang *et al.*, 2016)	KGE	TransR	Data-sparsity	- MovieLens - IntentBooks	Microsoft Satori	- Movie - Book
Entity2rec (Palumbo *et al.*, 2017)	GRL	Node2vec		MovieLens	DBpedia	- Movie
(Cai *et al.*, 2018)	GRL	Node2vec		- AAN[3] - DBLP[4]		- Citation
DKN (Wang *et al.*, 2018)	KGE	TransD		Bing News		News
CFKG (Zhang *et al.*, 2018)	KGE	TransE	Data sparsity	- Amazon		- CD - Clothing - Cell Phone - Beauty
HERec (Shi *et al.*, 2019)	GRL	Node2vec	Cold-starting	- Douban Movie - Douban Book - Yelp		- Movie - Book - Business

[3] http://clair.eecs.umich.edu/aan/index.php
[4] http://arnetminer.org/citation

N2VSCDNNR (Chen *et al.*, 2019)	GRL	Node2vec	Data-sparsity	- MovieLens - Yelp - Amazon		- Movie - Business - Musical Instruments
KTUP (Cao *et al.*, 2019)	KGE	TransH		- MovieLens - DBbook	DBpedia	- Movie - Book
KGCN (Wang *et al.*, 2019a)	DGRL	GCN	- Data sparsity - Cold-starting	- MovieLens - Book-Crossing - Last.FM	Microsoft Satori	- Movie - Book - Music
KGCN-LS (Wang *et al.*, 2019b)	DGRL	GCN	- Data sparsity - Cold-starting	- MovieLens - Book-Crossing - Last.FM - Dianping-Food[5]	Microsoft Satori[6]	- Movie - Book - Music - Restaurant
RecoGCN (Xu *et al.*, 2019)	DGRL	GCN		-Beidian		Social e-commerce
KGAT (Wang *et al.*, 2019)	DGRL	GAT	- Diversity - Explainability	- Amazon-book - Last-FM - Yelp2018	- Freebase	- Book - Music - Local businesses
HAGERec (Yang and Dong, 2020)	DGRL	GAT	- Explainability - Data-sparsity	- Last-FM - MovieLens - Book-Crossing		- Movie - Book - Music
KCAN (Tu *et al.*, 2021)	DGRL	GAT	Explainability	- MovieLens - Last.FM - Yelp		- Movie - Book

[5] https://www.dianping.com/
[6] https://searchengineland.com/library/bing/bing-satori

						- Local businesses
CGAT (Liu *et al.*, 2021)	DGRL	GAT		- Last-FM - MovieLens - Book-Crossing - Dianping Food		- Movie - Book - Music - Restaurant
KGIN (Wang *et al.*, 2021)	DGRL	GAT	Explainability	- Last-FM - MovieLens - Book-Crossing		- Movie - Book - Music
(Ragab and El-Kafrawy, 2022)	GRL	Node2vec	Cold-starting	- MovieLens	DBpedia	- Movie
(DKMN) (Ouyang and Zhou, 2022)	DGRL	GCN		- Yelp		Social e-commerce
KR-GCN (Ma *et al.*, 2022)	DGRL	GCN	Explainability	- Amazon-book - Last-FM - Yelp2018		- Books - Business - Music
(Shimizu *et al.*, 2022)	DGRL	GAT	Explainability	- ZOZOTOWN		- E-commerce
COAT (Dai *et al.*, 2022)	DGRL	GCN GAT		- MovieLens - Last.FM - Dianping-Food - Book-Crossing	- Freebase	- Movie - Book - Music - Restaurant

Conclusion

This chapter provided a comprehensive overview of RS, discussing their challenges, limitations, and various techniques for recommendation. Traditional RSs face constraints that hinder their ability to provide accurate recommendations. To address these challenges, a new type of RS based on Semantic Web technologies, such as ontologies and Knowledge Graphs, has gained popularity. These systems leverage the vast knowledge available on the Web of data by identifying relevant concepts from datasets and associating them with items to be recommended.

Furthermore, this chapter presented a comprehensive categorization of state-of-the-art semantic-aware recommender systems based on the employed embedding techniques. These techniques include Graph Embedding Methods, Knowledge Embedding Methods, and deep graph embedding methods.

The primary findings indicate that each category of work has its own strengths and limitations, particularly when dealing with Knowledge Graphs (KGs) due to their rich semantic nature. Additionally, it is observed that the issue of diversity has not been adequately addressed despite its importance in user satisfaction. Moreover, although state-of-the-art work has explored the explainability of recommendations, it primarily relies on path-based explanations rather than the generated embeddings. Thus, the contribution of embedding techniques in explainability remains unclear.

Overall, further research is needed to address the challenges in KG manipulation, enhance diversity in recommendations, and explore the potential of embedding techniques in improving explainability.

Chapter 3

From Knowledge Graphs to Personalized Recommendations, Leveraging Graph Embeddings For Improved Accuracy and Diversity

Contents

Introduction

The massive expansion of online data has led to information overload problem, where users are confronted with an overwhelming amount of choices that impedes their ability to make accurate decisions. Recommender systems (RS) intend to solve this problem by predicting users' preferences and recommending the most relevant items to each user. As stated in Chapter 2, Knowledge Graph-based Recommender Systems have proven the ability to produce more accurate recommendations. However, leveraging the heterogeneous and high-dimensional information encoded in KGs to improve recommendation is a challenging task. One feasible way to address this challenge, is to preprocess the KG by Graph Representation Learning techniques, which learn low-dimensional vector representations for nodes and/or edges. These representations minimize the engineering efforts, and can conduct to a higher predictive performance.

Most KG-aware RSs tend to focus on incorporating semantic knowledge into the original data, without ensuring that the semantics of predicates are maintained over the embedding process.

In this chapter, we tackle the subsequent research questions:

RQ 1. *How to incorporate the knowledge encoded in KGs into the user-item graph, while retaining the semantics conveyed by the predicates during the learning phase, with the goal of enhancing recommendation precision and promoting diversity in the recommended items?*

In pursuit of our main research objective, we develop four sub-research questions:

- **RQ 1.1.** *How to design a Knowledge Graph that aligns with both KGs and the user-item graph, while also promoting semantic preservation?*
- **RQ 1.2.** *How can a Recommender System leverage the efficacity of the Graph Representation Learning algorithm Node2vec in learning features from graphs?*
- **RQ 1.3** *How can translational methods generate item recommendations?*
- **RQ 1.4** *Is the proposed Knowledge Representation can insure the diversity of recommendations?*

The remaining sections of this chapter are structured as follows: Section 3.1 presents the proposed design of the Knowledge Graph, Section 3.2 describes the generation of latent embeddings using Node2vec, Section 3.3 outlines the application of the TransE method for item recommendation, Section 3.4 describes the diversification protocol, Section 3.5 provides details on the experimental setup, Section 3.6 presents the results of the comparative analysis between the proposed approach and both collaborative and content-based filtering systems, and finally, the chapter concludes with a summary.

Part of the work described in this chapter has been published in the proceedings of the 8th edition of the International Conference of Innovation and New Trends In Information Technology (INTIS) (Boughareb *et al.*, 2019).

3.1 Knowledge Graph Incorporation

In this subsection, we address **RQ 1.1.** *How to design a Knowledge Graph that aligns with both KGs and the user-item graph, while also promoting semantic preservation?*

To successfully leverage KGs for item recommendations, it's essential to design a representation model that effectively match the user-item graph with the data coming from the Linked Data cloud without semantic lost.

Before delving into the details of the Knowledge Graph construction phase, it is important to note that our work focuses on the movie recommendation domain as its primary application. Choosing the movie domain for recommendation can be a great option because movies are one of the most popular forms of entertainment. People of all ages, cultures, and backgrounds enjoy watching movies. The movie industry is also constantly growing and evolving, with new movies being released every week, which provides a large pool of data for recommendation models to work with. Movies are also quite subjective, which is a special quality that makes them a great fit for recommendation systems. Everyone has different tastes in movies, thus what one person may adore, another person may not even find enjoyable. Due to this subjectivity, personalized content-based recommendation systems can help users find movies they might not have otherwise known about. Moreover, movie recommendation systems are already widely used in popular platforms like Netflix, MovieLens, and Amazon Prime Video, and these platforms make data available for building recommendation systems. However, content-based

recommendation systems require additional features beyond what is typically provided in these data sets.

Knowledge Graphs can be the ideal solution to this problem, by combining the movie features with Linked data, it is possible to create a powerful content-based RS for movies that takes into account not only the basic features of the movie but also its context and related information. This can help to provide more accurate and personalized recommendations for users.

The process of movie-related data enrichment involves three steps. The first step involves extracting user-item data from the MovieLens dataset, which contains information about users and movie ratings. The next step is to link this data to entities in the Linked Open Data (LOD) cloud, specifically, DBpedia and LinkedMDB, which contains structured information about the movie domain. Once the data is linked to these external sources, additional information can be added to enrich the data. This could include information such as actors, directors, genres, release dates, and more. By leveraging the rich resources available in the LOD cloud, this process can help to create a more complete and accurate picture of the movie domain.

We highlight that while DBpedia contains a large amount of structured information extracted from Wikipedia, it may not have all movies, particularly those that are less popular or less well-known. Therefore, adopting a specific movie knowledge base like LinkedMDB may ensure that the most movie information is retrieved. LinkedMDB is a dedicated movie database that contains information on a wide range of movies, including both popular and less well-known titles. By using LinkedMDB in addition to DBpedia, we can increase chances of finding information on a wider range of movies.

The creation of our Knowledge Graph involved querying the DBpedia and LinkedMDB knowledge bases using the SPARQL query language in order to link them with the MovieLens dataset. By doing so, we were able to establish relationships and generate a Knowledge Graph that can be further explored and analyzed for recommendation purposes. First, we retrieve the title and the year of production of the current film from DBpedia for each movie item in MovieLens to create the matching movie object. We point out that the class we work with is dbo:Film from DBpedia. For instance, we submit a SPARQL query that provides as output the resource whose name matches the title of the target movie in order to acquire the mapping for

the film The Shawshank Redemption. The outcomes of the sample query are shown in Table 3.1.

Table 3.1. An example of SPARQL mapping for the movie The Shawshank Redemption. This query uses the dbo prefix to specify properties from DBpedia ontology and the lmdb prefix to specify properties from LinkedMDB. It matches the movie's title and release date in MovieLens with the corresponding URI of the same movie in the LOD cloud. It includes a filter to only retrieve information in English.

sparql query

```
<?xml version="1.0"?>

PREFIX rdfs: http://www.w3.org/2000/01/rdfschema#
PREFIX dbo: http://dbpedia.org/ontology/
PREFIX lmdb: http://data.linkedmdb.org/resource/movie/
SELECT DISTINCT ?title
WHERE {
    { SELECT DISTINCT ?film
    WHERE {
          ?film a dbo:film
          ?film a lmdb: film
                    }
      }
?film dbo:title ?title
?film dbo:release_date ? release_date.
FILTER (lang(?title) = 'en')
}
```

Result

http://data.linkedmdb.org/resource/**The_Shawshank_Redumption**

Next, we enrich the movie dataset with additional information related to the movie domain. We selected the most frequent properties to describe the movie content, which are dbo:starring, dbo:director, dbo:producer, dbo:distributor, dbo:writer and dct:subject. Table 3.2 illustrates the SPARQL query to populate the movie dataset with the relevant semantic properties.

Table 3.2 An example of SPARQL query for enriching the movie data. This query uses the same prefixes as the previous query to specify properties from DBpedia ontology and LinkedMDB. However, it retrieves information about the movie's subject, producer, actor, director, distributor, and writer. It also includes a filter to only retrieve information in English.

sparql query

```
PREFIX rdfs: http://www.w3.org/2000/01/rdfschema#
PREFIX dbo: http://dbpedia.org/ontology/
PREFIX lmdb: http://data.linkedmdb.org/resource/movie/
SELECT
(SAMPLE(?title) AS ?TITLE)
     (SAMPLE(?subject) AS ? SUBJECT)?
(SAMPLE(?director) AS ?DIRECTOR)
(SAMPLE(?distributor) AS ?DISTRIBUTOR)
(SAMPLE(?writer) AS ?WRITER)
```

```
(SAMPLE(?producer) AS ?PRODUCER)
(SAMPLE(?actor) AS ?ACTOR)
WHERE
{
?TITLE rdf:type http://dbpedia.org/ontology/Film.
?TITLE http://purl.org/dc/terms/subject ?subject.
?TITLE dbo:director ?director.
?TITLE dbo:distributor ?distributor.
?TITLE dbo:writer ?writer.
?TITLE dbp:producer ?producer.
?TITLE dbp:starring ?actor.
FILTER (lang(?title) = 'en')
 }
```

Those content features must be combined with the collaborative information to design the user-item interactions. For that we defined the property '*like(user, item)*' to model the fact that the user $u \in U$ likes the item $i \in I$. The property *'like'* depends on the explicit ratings, ranging from 1 to 5. We passed from 5-ratings-scale to a binary-rating-scale by assigning $r=1$ to ratings no less than 4 and 0 to other cases. The property *'like(user, item)'* will designed if and only if $r=1$.

As a result, a Knowledge Graph $KG=(V, T)$ has created. **(i)** V denotes the set of nodes representing users $U = \{u_1 , \ldots , u_n\}$, Items $I = \{i_1 , \ldots , i_m\}$ and different entities $E= \{e_1 , \ldots , e_k\}$ like actors, directors, and producers; **(ii)** T denotes the set of labeled edges represented as RDF triples . A portion of the generated movie Knowledge Graph created using the MovieLens and DBpedia datasets is shown in Figure 3.1.

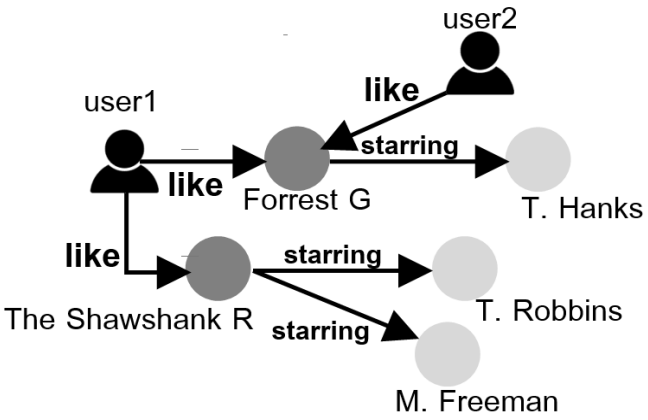


Figure 3.1. Knowledge Graph built by matching of DBpedia, LinkedMDB, and MovieLens. The graph differentiates items using the dark grey color and represents other entities in light grey. The representation includes *'like'* and *'starring'* properties as an example, but additional semantic properties are considered in the experiments.

Once the Knowledge Graph is created, it should be an input for an embedding algorithm in order to perform recommendation process. However, the embedding algorithms are not specifically designed to analyze semantic graph structures, which are characterized by their complex relationships often involving multiple layers of abstraction and context. Specifically, edges in KGs represent more than just a simple link between nodes, and analyzing these structures requires specialized algorithms and techniques that go beyond the scope of typical word embedding methods. Therefore, learning embeddings from a Knowledge Graph without finding a way to preserve the property information, leads to a great loss of semantics.

In order to support semantic information during the embedding process, we transform properties into entities by converting labeled edges into nodes. However, this mapping may alter the graph structure when applied to properties with complex connectivity patterns. To address this issue, we have developed an edge labeling function $f: T(KG) \rightarrow R$ that assigns a label to each RDF triple in the graph KG. The f function uses consecutive integers $\{1, ..., |N|\}$ to label the triples, where $|N|$ is the number of triples in the graph. Then, we transform each property to a pair of edges with a new node n_p, where the original edge-label will be labeled on the new node n_p, and the triple T becomes a pair of edges $< n_p, o >$. The pseudocode for the Property to Entity algorithm is provided in Table 3.3, and an illustration of the algorithm is given in Figure 3.2.

Table 3.3. Property to Entity algorithm pseudocode.

pseudo-algorithm

```
Input: KG (V, T): N triples T, M property p ⊂ T
Output: KG' (V', E)

1. for each triple T <s, p, o> 1..N
     f: T (KG)→ R

2. for each property p (1..M)
     create node np
     for i: 1.. N
     if p = np
         create edge <s, np> with label i
         create edge <np, o> with label i
```

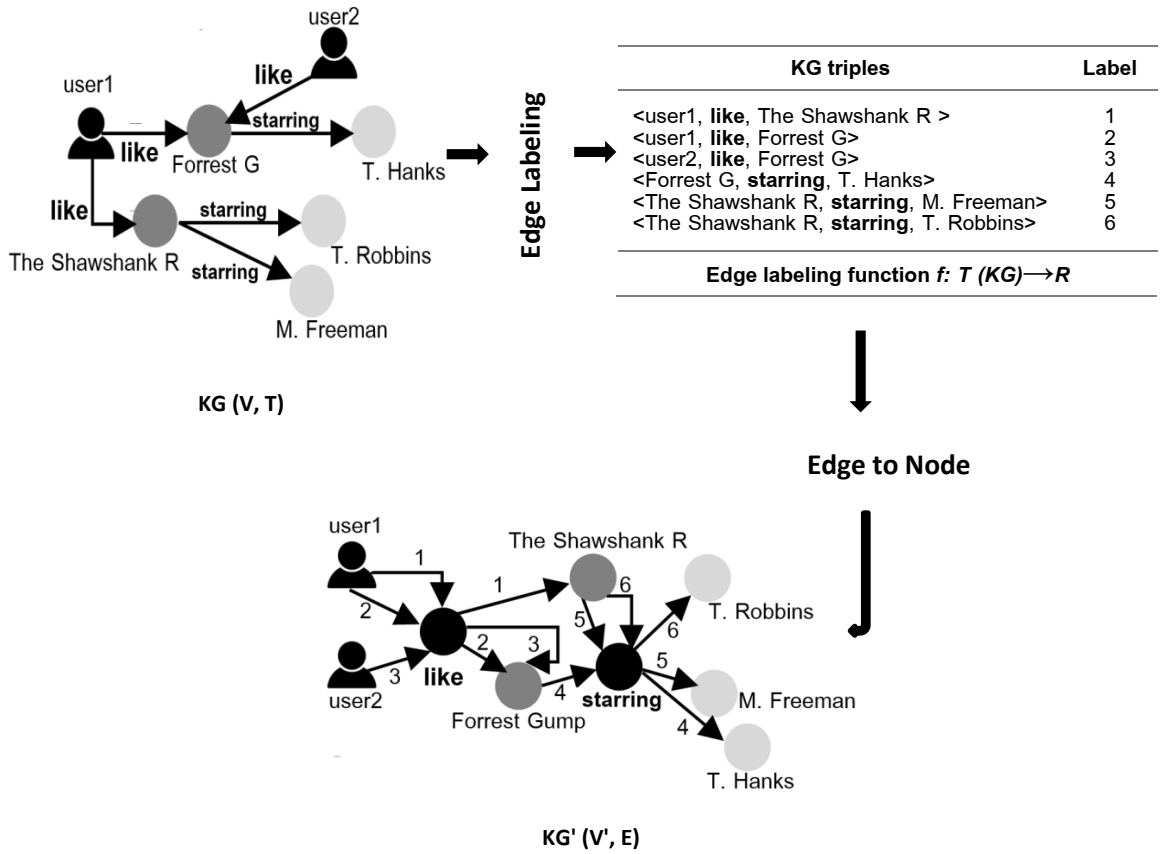


Figure 3.2. Illustration of the algorithm Property to Entity.

3.2 Knowledge Graph Embeddings Using Node2vec

In this subsection, we address **RQ 1.2:** *How can a Recommender System leverage the efficacity of Node2vec in learning features from graphs?*

3.2.1 Background

Node2vec (Grover and Leskovec, 2016) is a popular algorithm for learning embeddings of nodes in a graph. It uses a random walk approach to generate node sequences that capture the graph structure. One of the innovative features of Node2vec is its ability to incorporate an additional parameter to control the balance between exploring new areas of the graph and returning to areas already visited. Node2vec algorithm combines the biased random walk strategy with the Skip-gram model to learn node embeddings that capture the structural properties of the graph:

(i) Generate random walks: starting at a node, the algorithm performs a random walk of length L, where at each step it chooses a neighbor to visit next. However, instead of choosing the next node uniformly at random as in a standard random walk, Node2vec uses a biased random walk that is controlled by two parameters: p and q. The parameter p controls the likelihood of returning to the previous node, while the parameter q controls the likelihood of exploring nodes that are far away from the current node (see Figure 3.3).

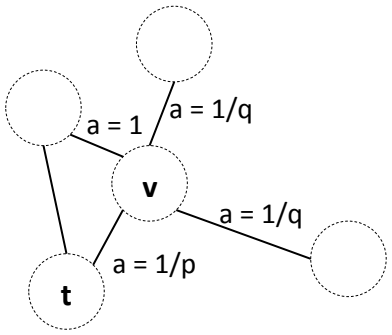


Figure 3.3 Transition probabilities for Node2vec's random walk. Specifically, it shows the probabilities when the walk is currently at node v and has just visited node *t*. These probabilities are determined by two parameters, *p* and *q*. A smaller value of p indicates a higher likelihood of the walk returning to the previous node, whereas a smaller value of q means that the walk is more likely to move one step away from the previous node. These probabilities can be adjusted by modifying the values of *p* and *q*, enabling the random walk to explore different paths and potentially achieve better results.

(ii) Learn embeddings: At a high level, Node2vec is based on the Skip-gram model (Collobert et *al.*, 2011), which is a popular neural network architecture for learning word embeddings. The Skip-gram model is trained to predict the context words (i.e., the words that appear near a given word) given a target word. The word embeddings are learned by minimizing the negative log-likelihood of the observed context words, given the target word. The Skip-gram model consists of an input layer, a hidden layer, and an output layer. The input layer is a one-hot encoding of the target node in the random walk, and the output layer is a *softmax* layer that predicts the context nodes that are likely to co-occur with the target node. The hidden layer is a low-dimensional vector representation, or embedding, of the target node. The dimensionality of the embedding is a hyperparameter that can be tuned to achieve the best performance on a given task (Figure 3.4).

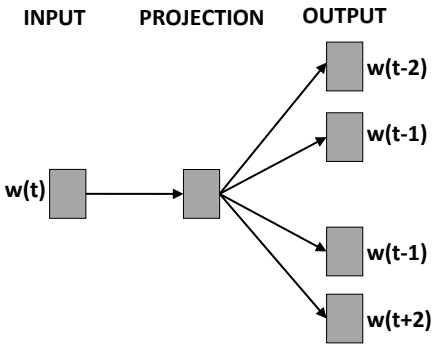


Figure 3.4. The Skip-Gram Architecture.

The Skip-gram model is trained using stochastic gradient descent to minimize the negative log-likelihood of the observed context nodes given the target nodes.

In the context of Node2vec, the Skip-gram model is applied to random walks on a graph instead of words in a sentence. Specifically, the input to the Skip-gram model is a sequence of nodes generated by a biased random walk on the graph. The goal of the model is to predict the nodes that are likely to co-occur with the current node in the random walk. To achieve this goal, the Skip-gram model learns two sets of embeddings: one set for the target nodes and another set for the context nodes. The target node embeddings are used to represent the current node in the random walk, and the context node embeddings are used to represent the nodes that are likely to co-occur with the current node.

More formally, given any graph $G = (V, E)$, Node2vec learns a mapping $f \leftarrow R^d$ for each node $v \in V$, which optimizes the following objective function:

$$\max_f \sum_{v \in V} log(P((N_s(v)|v)) \qquad (3.1)$$

Node2vec objective function maximizes the probability of observing a network neighborhood $Ns\ (v)$ for the node v, generated through a neighborhood sampling strategy S.

Inspired by the success of Node2vec algorithm, a number of researches attempted to use it to deal with Knowledge Graphs in order to perform recommendation task. For example, Entity2rec (Palumbo *et al.*, 2017) applied Node2vec to learn property-specific vector representation of nodes considering one property at the time. For each property they defined a sub-graph as the set of entities connected by a given property in order to capture different

semantic values of properties. This work differs with our approach that it aims to learn a global relatedness model that optimizes top-N item recommendations, while our approach aims at generating accurate and diverse recommendations based on user-property relatedness. Chen *et al.* proposed in (Chen *et al.*, 2019) a new clustering Recommender System based on Node2vec algorithm called N2VSCDNNR. The proposed approach showed a great potential to alleviate sparsity and cold-start problem. In (Xiaoyan *et al.*, 2018), the authors also utilized Node2vec algorithm to learn vector representations from a bibliographic Knowledge Graph. The embeddings used to perform a personalized citation recommendation. However, these works ignore the important role of semantic relations in user-item modeling. Movies can be connected based on their cast members rather than their themes, and they may have the same director but not the same writer. If we were to analyze the entire Knowledge Graph without considering the meaning of the properties, we would fail to capture these differences.

3.2.2 Node2vec for Node Embeddings

The Node2vec algorithm offers the flexibility to operate on both indexed and unindexed data. In our specific case, the input data is unindexed. To handle this, Node2vec indexes the nodes in the input edgelist, where each index represents an edge label. For instance, an unindexed edgelist may look like this:

```
node1 node2 1.0
node2 node7 1.0
```

In this format, the indexes represent the labels of the edges.

The supported input format for Node2vec is an edgelist structured as follows:

```
node1_id_int   node2_id_int   <weight_integer>
```

The output file consists of (number of nodes) multiplied by (numWalks) random paths. Each random path is represented as:

```
src_node_id_int        node1_id_int   node2_id_int   ...    noden_id_int
```

The supported input format for random paths is:

```
src_node_id_int        node1_id_int   ...    noden_id_int
```

The output files generated by node2vec are embeddings. The embeddings file follows the format:

```
node1_str       dim1 dim2 ... dimd
```

`dim1`, `dim2`, ..., `dimd` in the following representation indicate the d-dimensional representation learned by the neural network. Presented below is a small subset of the generated embeddings with a dimension of 5, where p = 0.4 and q = 1.0:

Node 4326	-0.5519583	-0.110089906	0.04873832	0.332927	-0.042838752
Node 1498	-0.2701256	0.20609294	0.11718224	0.26639208	-0.08950483
Node 2011	-0.37827548	0.053858332	0.2333144	0.27054447	-0.25751817
Node 1299	0.051720805	-0.029679134	0.18991205	0.38948807	0.011968189
Node 7815	-0.5231582	0.1241182	0.041150503	0.042417932	0.025797702
Node 1694	-0.31628376	0.124384426	0.07866778	0.38339454	-0.21936557
Node 1722	-0.40695754	-0.2626987	0.21833058	0.41458955	-0.001674862
Node 5308	-0.30351567	0.09386052	-0.22476502	0.20128608	0.036622655
Node 2916	-0.3778163	0.049850017	0.14839733	0.22435383	-0.25909367
Node 3824	-0.40181482	0.029769193	-0.00677491	0.38215357	-0.10893722

3.3 Item Recommendation

In this subsection, we address **RQ 1.3** *How can translational methods generate item recommendations?*

The item recommendation problem consists of ranking a list of candidate items *N* according to user preferences. This involves assigning a score to each user-item pair *(u, i)* to generate a subset of relevant items $M \subset N$ for the user *u*. We consider two different strategies to recommend items:

- **Recommendation with TransE score function**

TransE is an embedding method for Knowledge Graphs, it learns representations of entities and relations so that $s + p \approx o$ where is a RDF triple. TransE uses the following score function: $f(s, p, o) = d(s + p, o)$, where d is a distance function. For a triple, if the score $f(s, p, o)$ is closer to zero, the triple is considered as true. In our approach, we assign a score for each triple, where $u \in U$ is a user, $i \in I$ is an item (the items here are movies). As an example, for the triple , the TransE score function uses the vectors for the three elements in the triple as follows: $f(user1, like, Titanic) = d(user + like - Titanic)$ If $f\ user1, like, Titanic \approx 0$ then the film Titanic

is not relevant for the $user1$ and we will not be recommended for him, else i.e., $f\ user1\ ,like, Titanic \approx 1$, we recommend Titanic for $user1$. We note that the cosine distance is adopted.

- **Recommendation using cosine similarity**

We define the user-item relatedness score as the cosine similarity between their vector representations the cosine measure is defined by Equation 3.2.

$$rel(u,i) = \frac{\vec{u} \cdot \vec{i}}{|\vec{u}| \cdot |\vec{i}|} \qquad (3.2)$$

If $rel(u,i)$ is closer to zero, the item *i* will considered as relevant and will be recommended to the user u. Recommending items similar to the ones the user preferred before can achieve higher precision. However, users tend to be more satisfied with varying recommendations, i.e. being exposed to a content, which can promote discovery of something unexpected.

3.4 Item Diversification

Within this subsection, we will focus on answering research question **RQ 1.4.** *Is the proposed Knowledge Representation can insure the diversity of recommendations?*

To address the diversity problem, we proceed to use property embeddings in order to understand the user tastes and determine how he chooses a movie to watch. By measuring user-property relatedness, we can determine which property (i.e., movie characteristic like actors, directors and movie genres) influences the user's choice. For example, one can select a movie according to its genre, another selects a movie according to their favorite actors, also, some users enjoy movies that have been directed by a certain director or produced by a particular producer. Hence, considering these points allows controlling the diversification protocol. First, we define the user-property relatedness score as the cosine similarity between their vector representations. We consider only content properties which are dbo:starring, dbo:director, dbo:producer, dbo:distributor, dbo:writer and dct:subject. Then we rank the results in ascending order. Hence, for each user, we have six order preferences from lower to higher. Next, we select *k* items from the recommended-item list (RI-list). We process now the list from position *k+1*. For each movie m in RI-list, we measure its relatedness with each property p. According to this distance we add m to the corresponding property order.

3.5 Experimental Setup

This section outlines the experimental setup, encompassing details about the datasets used in the experiments, the process of configuring, and how the system was assessed.

3.5.1 Datasets

In our study, we utilized three datasets to enhance our understanding and recommendations in the movie domain. Two of these datasets are from the Linked Open Data (LOD) cloud, which provides a vast collection of interlinked and semantically rich data. The first dataset, LinkedMDB, is specifically focused on movies. The second dataset, DBpedia, is a cross-domain knowledge base that covers various domains, including movies. The third dataset utilized in our study is the benchmark dataset called MovieLens, which is widely used for movie RS. Table 3.4 presents the statistics for the underlined datasets.

- MovieLens. This benchmark dataset contains 100,000 explicit ratings which are made on a 5-star scale, 943 users, and 1,682 movies.

- DBpedia. Is a cross-domain dataset in the Linked Data cloud. In our work, we considered the DBpedia class dbo:Film which represents the movie domain. Then, we selected intuitively the most frequent properties used to describe the movie content which are dbo:starring, dbo:director, dbo:producer, dbo:distributor, dbo:writer and dct:subject.

- LinkedMDB. Is a dataset in RDF format that is focused on movies, and it contains around 94,000 movies along with associated metadata such as actors, directors, and genres.

Table 3.4. Statistics of the datasets.

Dataset	Type	Domain	Users	Items
MovieLens	Structured	Movie	943	1682
DBpedia	RDF	Cross domain	/	/
LinkedMDB	RDF	Movie	/	94, 000

3.5.2 Evaluation Protocol

We split the data into a train set (80%) and a test set (20%). The first set used to learn embeddings from the Knowledge Graph, when the second set used to test the performance of recommendation system.

The evaluation was made based on:

- Accuracy using precision metric (P@N) computed using the Equation 3.3 with two list sizes.

$$Precision@N(u) = \frac{number\ of\ relevant\ items\ in\ the\ topN\ list}{N} \quad (3.3)$$

- Diversity using Intra-list diversity measure (ILD@N) computed using the Equation 3.4.

$$ILD@N(u) = \frac{\sum_{i=1}^{n} \sum_{j=i}^{n} \left(1 - similarity(c_i, c_j)\right)}{N * (N-1)/2} \quad (3.4)$$

Where $c_i \dots c_n$ are items in a set of recommendation list and N is the recommended list.

We compared our system with three representative state of the art recommenders:

- Content-Based (CB). CB algorithm utilize the characteristics of items or users to make personalized recommendations. These systems analyze the content or properties of items and match them with the user's preferences or historical behavior to suggest items that are similar or relevant to their interests.

- Item Based K-Nearest Neighbor (I-KNN). Is a widely used CF algorithm. Its primary objective is to predict the rating of a specific item by leveraging the ratings of similar items provided by users. The I-KNN algorithm operates by first identifying *K* items that are most similar to the target item. The similarity is typically calculated using a similarity metric such as cosine similarity. Once the *K* most similar items are determined, the algorithm infers the rating of the target item based on the user's ratings on those similar items.

- Matrix Factorization Algorithm (MF). Is a technique used in RS that decomposes a user-item rating matrix into user and item factor matrices. The algorithm learns the factor matrices by minimizing the error between predicted and actual ratings.

3.5.3 Configuration

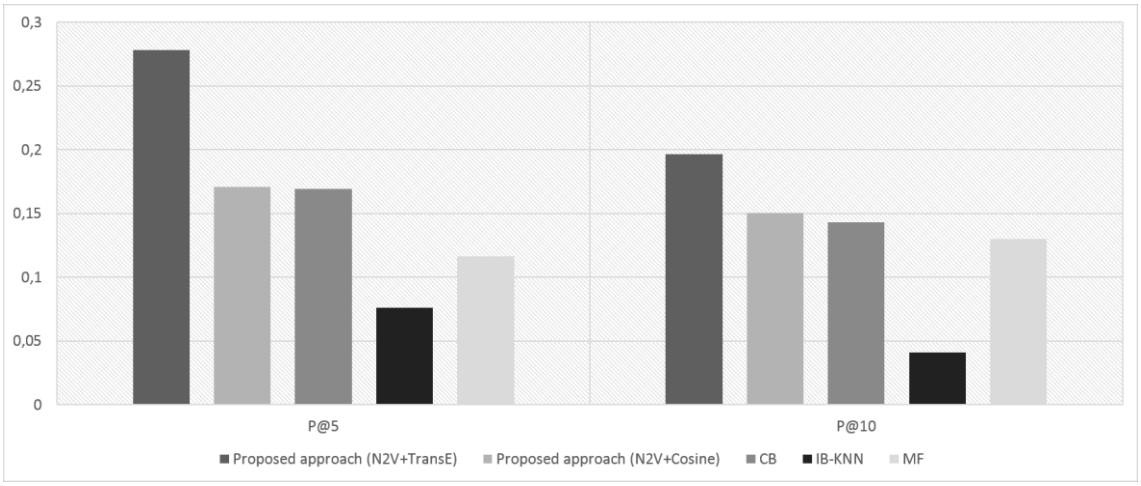
We used the following settings for Node2vec parameters:

- walk length=10: The length of walk for each node.
- num walks=10: The number of walks per node.
- p=q=1: We followed the same grid search than (Grover and Leskovec, 2016) to choose the best values of two step random parameters p and q.
- dimension size d=10: we leave the default dimension size of embedded vector.

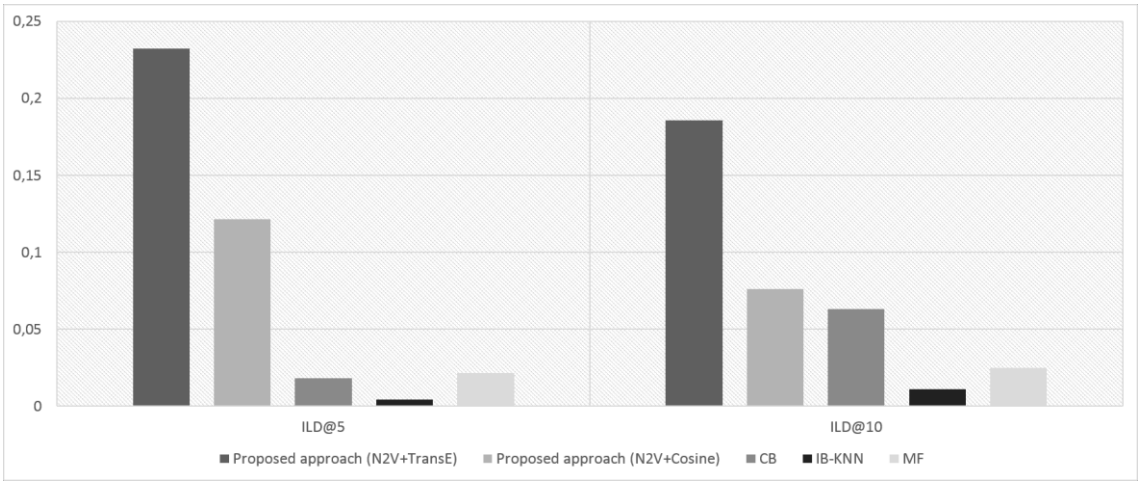
3.6. Experimental Results

3.6.1 Results and Discussions

The results presented in Figure 3.5 (a) and (b) showcase the performance of each approach in terms of precision (P@k) and diversity (ILD@k) metrics, respectively. In these metrics, the value of k denotes the top k recommendations generated by each approach.



(a)



(b)

Figure 3.5. Performance of the proposed approach against the baselines.

Firstly, the proposed approach using N2V+TransE demonstrated the highest precision values at both top 5 (P@5) and top 10 (P@10) recommendations. This suggests that this approach effectively identified relevant items that matched users' preferences. Additionally, the high item coverage at both ILD@5 and ILD@10 indicates that the proposed approach was able to provide a broad range of diverse recommendations, covering a significant portion of the item space. On the other hand, the proposed approach using N2V+Cosine achieved moderate precision values, although slightly lower than the N2V+TransE approach. However, it showed relatively lower item coverage, suggesting that the recommendations may have been more focused and less diverse compared to the N2V+TransE approach.

Comparing the proposed approaches with the Content-Based (CB) approach, it is evident that the proposed approaches outperformed the CB approach in terms of precision. However, the CB approach exhibited the lowest item coverage values, indicating that it may have recommended a limited set of items that matched the users' content preferences.

The Item-Based KNN (IB-KNN) approach yielded the lowest precision and item coverage among the evaluated methods. This suggests that the IB-KNN approach may have faced challenges in accurately identifying and recommending items that align with users' preferences. Further investigation is needed to understand the reasons behind its relatively poor performance.

Lastly, the Matrix Factorization (MF) approach showed moderate precision and item

coverage values, indicating its potential in providing reasonable recommendations. However, it is important to note that its performance was surpassed by the proposed approaches using N2V+TransE and N2V+Cosine.

Upon analyzing the results and performance differences among the evaluated approaches, several factors may have contributed to the observed variations. The choice of data representation techniques, such as N2V (Node2vec), TransE, and cosine similarity, played a crucial role in the performance differences. The proposed approach utilizing N2V+TransE achieved higher precision and item coverage, indicating the effectiveness of embedding-based representations and the TransE model in capturing meaningful relationships and similarities between movie items. The variations in performance could also be attributed to the availability and completeness of data across the integration of LinkedMDB and DBpedia datasets. By leveraging the strengths of both datasets, the RS can benefit from a more complete and diverse set of movie properties, leading to enhanced accuracy and recommendation quality. The combination of different movie properties from LinkedMDB and DBpedia allows for a more comprehensive understanding of user preferences and movie characteristics. This integration not only enhances accuracy by providing more relevant recommendations but also ensures diversity by considering a wider range of movie properties.

The final and arguably most crucial factor influencing the performance of the recommender systems is the effective modeling of user preferences, which stands as the primary contribution of this study. By incorporating semantic information and ensuring its preservation throughout the recommendation process, the study has achieved highly informative output embeddings and a remarkable richness in the recommendations. This integration of semantic information has significantly contributed to enhancing the accuracy of the recommender systems, as it enables a deeper understanding of user preferences and their underlying semantic relationships with movie properties.

Moreover, the incorporation of semantic information has also played a vital role in ensuring diversity in the recommended items. By leveraging the semantic connections between different movie properties, the recommender systems can provide a wider range of diverse recommendations that align with users' individual preferences. This emphasis on diversity is essential for catering to the heterogeneous tastes and interests of users, enhancing their overall satisfaction with the recommendations.

3.6.2 Computational Complexity

The computational complexity of the proposed systems depends on two factors: the number of nodes V' and the number of users U. The training process can be described as the sum of two components: the time it takes for Node2vec to generate the embeddings, and the time it takes to calculate the user-item relatedness O(1). The complexity of Node2vec can be broken down into two main parts:

- Generating the random walks: The time complexity of generating random walks for a graph with N nodes and E edges is O(d * N * t), where d is the length of the random walk, and t is the number of random walks generated per node.

- Optimizing the embeddings: The time complexity of optimizing the embeddings using stochastic gradient descent (SGD) depends on the number of iterations, the batch size, and the dimensionality of the embeddings. In general, the time complexity of SGD is O(I * B * D), where I is the number of iterations, B is the batch size, and D is the dimensionality of the embeddings. Therefore, the overall time complexity of Node2vec can be expressed as O(d * N * t + I * B * D), where N is the number of nodes in the graph. The time for testing is: O(VI). Therefore, the total training complexity is linear : O(VI)

Conclusion

In this chapter, we investigated Node2vec framework in the movie recommendation domain. The key idea was to modify the graph structure in order to learn low-dimensional embeddings for both nodes (i.e., entities) and edges (i.e., properties). First, we reconstructed the KG built from DBpedia and MovieLens, by transforming property edges into entity nodes. Next, we applied Node2vec algorithm to generate embeddings for each entity in the graph. Incorporating properties into the embedding process allowed to understand user preferences and then tackle the diversity problem. We performed recommendations by testing two methods. The first inspired by TransE method. The second consisted on measuring the user-item relatedness using cosine similarity. The results showed the best performance in terms of precision and diversity.

To further enhance recommendation performance, several potential avenues can be explored. Firstly, investigating more effective data representation techniques that leverage the implications of KGs in user preference modeling can be valuable. Secondly, given the success

of advanced deep learning-based approaches in capturing complex patterns and relationships in graph-structured data, their utilization has become crucial. Furthermore, the combination of KGs and advanced deep learning techniques could not only enhance the accuracy and personalization of recommender systems but also open up avenues for explainability.

Chapter 4

Explain-KGCN: A convolutional weighted Knowledge Graph for explainable recommendation

Content

Introduction

The use of Knowledge Graphs in Recommender Systems has shown great potential for improving the accuracy and relevance of recommendations. By incorporating a semantic layer that captures relationships between entities, it is possible to provide more personalized and context-aware recommendations that better reflect the user's interests and preferences.

At the same time, the emergence of mobile devices and social media platforms has transformed the way users interact with technology and with each other. As a result, users are now more expressive than ever before, sharing their opinions, preferences, and experiences with others online. This has led to the generation of more complex data that can be difficult to analyze and understand using traditional techniques.

However, designing a KG-based RS is not without its challenges. One major challenge is the effective representation of knowledge. A well-designed representation should capture both the semantic relationships between entities in the KG as well as the preferences and interests of users. Once an effective representation is established, the next challenge is to use advanced ML techniques that can effectively process and analyze complex data. This is where Deep Graph Representation Learning methods come in. Graph Convolutional Networks (GCNs) are a class of Graph Neural Networks (GNNs) algorithms that can operate directly on graph-structured data. They have shown great promise in learning from complex graph structures and can be used to develop KG-based RSs that can provide more personalized and accurate recommendations.

In this chapter, we tackle these research questions:

- **RQ1.1:** *How can we create a knowledge representation that melds Linked Data with intricate user-item graph preferences while preserving semantic information?*
- **RQ1.2** *How can Knowledge Graphs (KGs) and user-item interactions be included into Graph Convolutional Networks (GCNs) to produce recommendations that are not only precise but also interpretable and in line with semantics?*
- **RQ1.3** *How negative and positive users' feedbacks affect the prediction performances?*
- **RQ1.4** *How does the proposed metric contribute to preserving semantics and enhancing the system accuracy?*

- **RQ1.5** *Is the proposed system can insure the explainability of recommendations?*

The remaining sections of this chapter offer a comprehensive exploration of the proposed approach. The chapter begins with a background (Section 4.1) to provide the necessary context for the research. Section 4.2 delves into the Explain-KGCN approach, starting with the construction of the Knowledge Graph (Section 4.2.1). The chapter then explores the process of weighting the KG (Section 4.2.2) and embedding it using Graph Convolutional Networks (GCNs) (Section 4.2.3). Section 4.2.4 discusses the item recommendation process using the Explain-KGCN approach, while Section 4.2.5 focuses on making recommendations explainable. In Section 4.3, an experimental evaluation is conducted to assess the performance of the Explain-KGCN approach. This section begins with the experimental settings (Section 4.3.1) and presents comparative results (Section 4.3.2) to evaluate the proposed approach against other methods. Furthermore, the explainability aspect of the Explain-KGCN approach is analyzed (Section 4.3.3). The chapter also explores the impact of the DC_r factor (Section 4.3.4) and provides a case study (Section 4.3.5) to further validate the approach. The chapter concludes with a summary and conclusion, highlighting the key findings and contributions of the research.

Part of the work described in this chapter has been published in 5th issue of the 21st volume of the International Journal of Information & Knowledge Management (JIKM) (Boughareb *et al.*, 2023).

4.1 Background

Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017): are a type of Neural Network designed to operate on graph-structured data and aim to learn a node-level representation that captures both the local and global structures of the graph. The GCN architecture is based on a Convolutional Neural Network (CNN) (LeCun and Bengio, 1998) applied to the graph. The main idea is to perform a linear transformation of the node features using the adjacency matrix, which describes the connections between nodes in the graph. This operation is similar to a convolutional operation in a traditional CNN, where a filter is applied to each pixel of an image to generate a feature map.

The core component of GCN is an aggregator function that learns a low-dimensional representation for each node in the graph by aggregating information from its neighbors. More

formally, in a GCN with K layers, its k^{th} layer gets a feature vector for each node i from the previous layer $(k-1)$, and transforms it by aggregating neighborhood information $N(i)$ of node i with fixed weights $w_{i,j}^{l}$.The final embedding $\overrightarrow{h_i^l}$ of node i is generated by performing recursive neighborhood aggregation across multiple hidden convolutional layers. The embedding learning process by the k^{th} layer is generally expressed by Equation 4.1 and Equation 4.2, respectively:

$$\overrightarrow{h_{N(i)}^{k}} \leftarrow TRANSFORM^{k}\left[AGGREGATE\left[\left\{\left(\vec{h}_j^{k-1}, w_{i,j}^{k}\right) | j \in N(i)\right\}\right]\right] \quad (4.1)$$

$$\overrightarrow{h_i^k} \leftarrow COMBINE\left(\vec{h}_j^{k-1}, \overrightarrow{h_{N(i)}^{k}}\right) \quad (4.2)$$

4.2 Item Recommendation Based on Weighted KG and GCNs (Explain-KGCN)

The architecture of Explain-KGCN is depicted in Figure 4.1, showcasing its components and flow. The system consists of four main steps: (1) KG construction, which aims to incorporate rich side semantic information to enhance recommendations; (2) KG weighting, in which each node's importance is measured; (3) KG embeddings with GCNs, which aims to generate for each given entity a set of relation-specific embeddings that depend on each semantic relation in the KG; (4) Top-N explained item recommendation, which aims to provides an explainable ranked list of items that each user is likely to be interested in. The steps are explained in detail in the following subsections respectively.

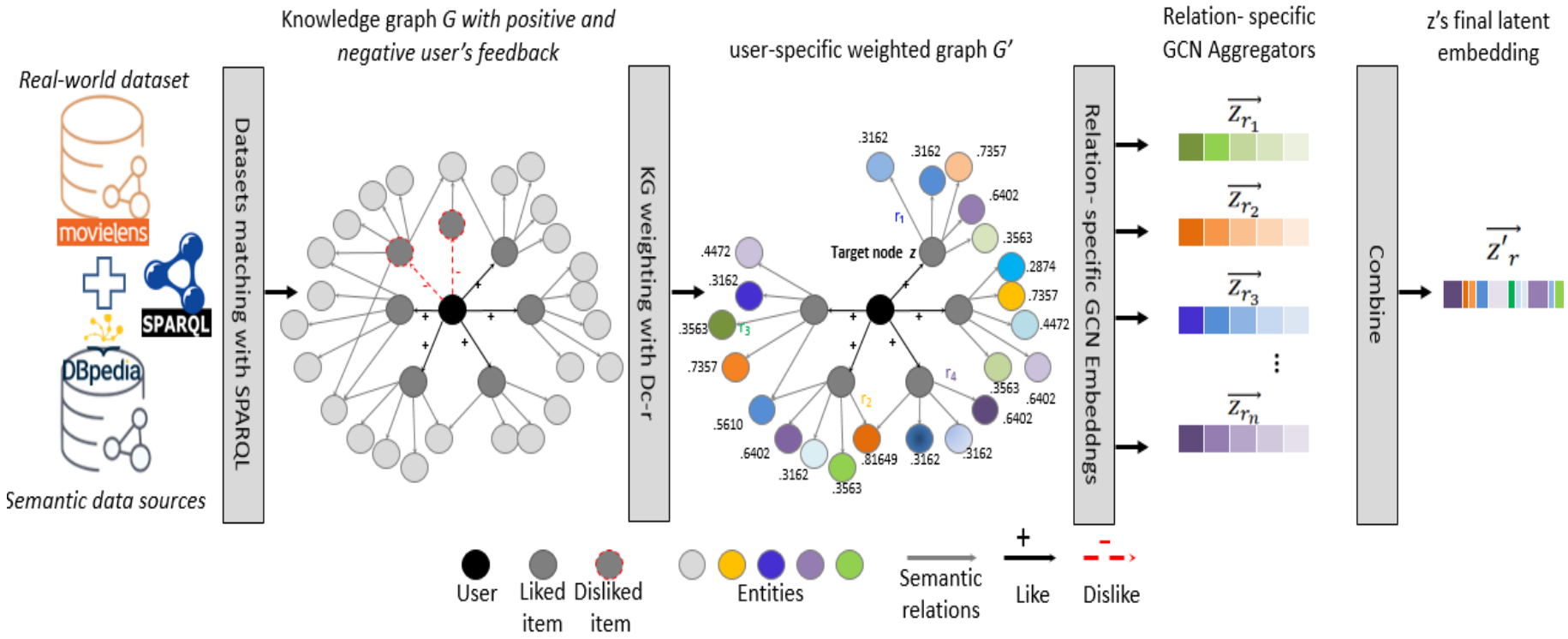


Figure 4.5. The overall architecture of Explain-KGCN.

4.2.1 Knowledge Graph Construction

In the first part of this subsection, we address **RQ1.1:** *How can we create a knowledge representation that melds Linked Data with intricate user-item graph preferences while preserving semantic information?*

In our previous contribution, as described in Chapter 3 Section 3.2, we established connections between the MovieLens dataset and the DBpedia knowledge base to create a Knowledge Graph. By querying DBpedia using the SPARQL query language, we matched each item in the dataset to its corresponding entity in DBpedia, retrieving essential information such as the title and production year of the target item. It is important to note that in addition to the MovieLens dataset, we also utilized the LibraryThing dataset for book recommendation purposes. The same procedure was applied to establish connections between the books in the dataset and their corresponding entities in DBpedia. We focused on processing the DBpedia classes dbo:Film and dbo:Book. Once the datasets and the DBpedia knowledge base have been connected, we enriched the movie and book datasets with structural knowledge specific to each domain. For the movie dataset, we selected several 1-hop properties such as genres, actors, directors, producers, and writers to describe the movie content. Similarly, for the book dataset, we chose properties including genre, author, publication date, and language to describe the book entities.

It is important to emphasize that the feedback we are dealing with is explicit in nature. This means that we obtained user-item interactions directly from the user's evaluation of items, such as providing 5-star ratings or indicating likes/dislikes. This type of feedback can be categorized into explicit positive feedback and explicit negative feedback. Explicit positive feedback refers to instances where the user expresses liking for an item (in a like/dislike system) or provides a favorable evaluation (e.g., rating >= 3 in a 5-star rating scenario). On the other hand, explicit negative feedback occurs when the user dislikes an item or gives it a low rating.

In many real-world applications, existing user-item interaction models often focus solely on explicit positive feedback while ignoring explicit negative feedback. This means that they primarily consider what users like without taking into account their dislikes. However, by incorporating both positive and negative explicit user feedback, our proposed model leverages richer information about user preferences. It is worth noting that negative feedback can be challenging to utilize effectively due to the prevalence of negative ratings and potential

discrepancies between users' negative comments and their evaluations of specific item features. For example, disliking a movie does not necessarily imply disliking its lead actor or director. To address this challenge, we designed an initial semantic graph representation that captures all evaluated features in a comprehensive manner, enabling us to effectively leverage negative evaluations in our model.

The data used for generating recommendations typically consists of user-item interactions, where users $i \in U$ and items $j \in I$ with a response $Y_{ij} \in Y$. We define the collaborative property '*like (user, item)*' to model the fact that the user $u \in U$ likes the item $i \in I$ as <*user1, like, The last castle*>. Additionally, we introduced the property '*dislike (user, item)*' to represent the user's dislike for an item. The '*like*' property captures explicit positive ratings, which range from 1 to 5. To convert the ratings to a binary scale, we assigned a value of 1 to ratings that were 4 or higher, and 0 to other cases. The '*like (user, item)*' property is created only when the rating is 1. This process results in the creation of a Knowledge Graph, $KG = (V, T)$, where V represents the nodes representing users $U = \{u_1, \ldots, u_n\}$, items $I = \{i_1, \ldots, i_m\}$, and other entities $E = \{e_1, \ldots, e_k\}$ such as actors, directors, and producers. T represents the labeled edges in the form of RDF triples <*s, p, o*>.

4.2.2 Knowledge Graph Weighting

A KG is a heterogeneous digraph as it encompasses various types of entities interconnected by multiple relationships. In the context of KG-aware RS networks, there are typically three types of nodes representing users, items, and semantic entities (e.g., actors, directors, genres, etc.). The specific semantic relationships between nodes depend on the recommendation domain, such as "starring," "produced-by," or "has-genre." For instance, in a movie KG-based RS, triples like <*User1, Item1, like*> and <*Item1, Entity1, director*> would exist. Different node types exert varying influences on the network, necessitating distinct weights to measure their importance relative to the information they provide.

Weighting the semantic graph in KG-based recommendations involves crucial considerations. Firstly, the weights should be personalized to reflect individual user preferences. For example, an actor entity may be liked by one user u_1 but not by another user u_2, indicating the need for different weights based on the user who expressed the preference. Secondly, the weights should capture the semantics encoded within the various relations of the

heterogeneous network. Lastly, the importance of a node should reflect its topological role in the entire graph.

Most KG-based RS models treat each entity equally, operating on an unweighted KG with one-hot encoding for node inputs. However, this approach overlooks the significant differences in the contribution of each entity to the overall information content. To address this, we propose a novel relational weighting metric called DC_r, inspired by the Degree Centrality measure defined in Equation 4.3.

$$d_i(e) = \frac{d(e)}{n-1} \tag{4.3}$$

In graph theory, the degree centrality of a node refers to the number of edges connected to that node, representing the count of its neighbors. DC serves as a measure to determine the importance of a node e within the network. Nodes with the highest degree centrality are considered the most important since they receive a significant amount of connections from other nodes, indicating their influence.

It's important to note that in directed graphs like KGs, there are two distinct measures of degree centrality: in-degree and out-degree. In-degree centrality calculates the number of incoming edges to a node, while out-degree centrality represents the number of outgoing edges from a node. These measures provide insights into the flow of connections within the graph, capturing the influence a node receives from others (in-degree) and the influence it exerts on other nodes (out-degree).

DC_r aims to assess the significance of each node in the KG by estimating its importance. Our choice to utilize the DC metric is driven by the following motivations: (i) DC represents a straightforward centrality measure based on the notion that nodes with more connections hold greater importance within the network. (ii) DC focuses solely on direct neighbors or one-hop connections, allowing us to infer the influence of immediate neighbors on the computation of the current node's weight. This becomes particularly relevant in scenarios where a hierarchical semantic topology exists, as is often the case in Knowledge Graphs. The specific formulation of the proposed metric is presented in Equation 4.4.

$$DC_r(e) = \frac{\left(DC^{in^+}\right)^{DC^{in-}}}{\sqrt[\alpha]{N_r - 1}} \tag{4.4}$$

where, DC^{in^+} and DC^{in^-} denote the number of edges from positive and negative feedbacks, respectively, N_r represents the total number of nodes sharing the same edge type as e, α is a normalized parameter represented experimentally.

The DC_r metric takes into account three factors that determine the personalized importance of each entity node in the graph: (1) the graph's topology, as inspired by the degree centrality measure; (2) the semantic nature of the KG, where the weight of an entity depends on its semantic type and the neighboring nodes of the same type; and (3) the personalized aspect, achieved by considering users' explicit positive and negative feedback to weight the nodes in the graph. To illustrate the concept of assigning weights to edges, with the target edge being the director property. Table 4.1 demonstrates how different weights are assigned to the director entity Steve Oedekerk, even though both user-2 and user-163 like the same movie (item-19).

Table 4.1. An example of DC_r weight computing.

	Director	UserId	MovieId	DC_r Weight
0	director-Steve Oedekerk	user-2	item-19	0.262613
1	director-Steve Oedekerk	user-159	item-19	0.267261
2	director-Steve Oedekerk	user-163	item-19	0.135147
3	director-Steve Oedekerk	user-196	item-19	0.210819
4	director-Steve Oedekerk	user-379	item-19	0.632456
	…	…	…	…

4.2.3 Knowledge Graph Embedding with GCNs

In this section, we address the research question **RQ1.2:** *How can Knowledge Graphs (KGs) and user-item interactions be included into Graph Convolutional Networks (GCNs) to produce recommendations that are not only precise but also interpretable and in line with semantics?*

To answer this question, we propose leveraging GCNs as a powerful framework for recommendation systems. The main objective is to incorporate relation types during the aggregation process to effectively model the semantics. The procedure is detailed in Algorithm 4.1, which describes the learning steps and computations involved.

Algorithm 4.1: Relation-specific GCN

Inputs: $\overrightarrow{z_i}$: input feature vector of the target node i,
$\{\vec{z}_e^r | e \in N(i)\}$: input relation-specific representation vectors of i's neighbors.

Outputs: $\overrightarrow{z_i}'$: global i's embeddings,
$\overrightarrow{z_i^r}'$: relation-specific embeddings of i.

Begin

//Transform neighbors representations (multiplication with shared weight).

//aggregate the resulting vectors (softmax inner product).

$\boldsymbol{c_e^r} = Aggregate(\{Transform(\overrightarrow{z_i}, \vec{z}_e^r) | e \in N(i)\}, w);$

$\boldsymbol{\vec{h}_e^r} = \sum(c_e^r \cdot \vec{z}_e^r);$

//Save intermediate relation-specific embeddings.

$\boldsymbol{\vec{h}_e^{r\prime}} = Save(\vec{h}_e^r);$

//Concatenate the resulting i's embeddings through a fully connected and biased layer with it's initial feature vector.

$\boldsymbol{\overrightarrow{z_i}'} = ReLU(\overrightarrow{z_i} || \vec{h}_e^r)$.

End

Algorithm 4.1, called "relation-specific GCN," takes as input the embedding of the target node i, the embeddings of i's relation-based neighbors, and the neighbor weights. The output is the updated embedding of the target node and relation-specific embeddings for each node.

The algorithm starts by initializing the node embeddings as node feature vectors. It then utilizes neural networks to transform and aggregate the features on the graph, ultimately computing the node embeddings. The node embeddings depend on the current embedding of the target node and the embeddings of nodes in its r-neighborhood, represented as $\{\vec{h}_e^r \mid \forall e \in N_r\ (v)\}$, where $N_r\ (v)$ denotes the neighbors of node n based on relation r.

The algorithm iteratively updates the node embeddings using relation-aware aggregators. The features of nodes located in the l-hops neighborhood of the target nodes are efficiently aggregated by recursively applying the aggregators.

During each iteration, the target node's neighbors' representations undergo a transformation (ReLU activation) before applying the aggregator function γ on the resulting set of u neighbor's embeddings (line 1). This step produces the aggregated neighborhood vector e_u. The resulting

embedding is then combined with the current representation of node u (Line 2), considering the importance of neighbors and relations in the aggregation process.

The algorithm outputs the final embedding of the target node (Line 4) and a set of relation-specific embeddings for each node (Line 3). This enables the model to capture the semantics encoded in the relation types while aggregating the neighbor vector representations.

4.2.4 Item Recommendation

The item recommendation problem provides a ranked list of items that user u is likely to be interested in. Relying on the final outputs of our proposed GCN model, each item i is represented as a global vector $\overrightarrow{z_i}'$. The relatedness between each candidate item i' and the items previously liked by u is computed to: (i) decide if the item i' could be recommended to the user or not and then sort a list of recommended items, (ii) rank the sorted list in descending order, with the most similar item on top. The cosine similarity between each vector $\overrightarrow{z_i}'$ in u's profile and the candidate item vector $\overrightarrow{z_{i'}}'$ is measured by Equation 4.5 as follows:

$$sim_{cosine}\left(\overrightarrow{z_i}', \overrightarrow{z_{i'}}'\right) = \frac{\overrightarrow{z_i}' . \overrightarrow{z_{i'}}'}{\left\|\overrightarrow{z_i}'\right\| \left\|\overrightarrow{z_{i'}}'\right\|} \qquad (4.5)$$

Cosine similarity is a metric used to measure how similar two items are. It measures the cosine of the angle between vectors $\vec{i}$ and $\vec{i}'$. The output value ranges from 0–1. If $sim\left(\overrightarrow{z_i}', \overrightarrow{z_{i'}}'\right)$ is closer to one, the item i' will considered as relevant and will be recommended to the user u, where as if $sim\left(\overrightarrow{z_i}', \overrightarrow{z_{i'}}'\right)$ is closer to zero, the two vectors have no match then, i' will not be recommended. By means of the computed similarities, for each user u, the corresponding items $\{i'_n\}$ are ranked in such a way similar or close items to the user profile i.e. $sim\left(\overrightarrow{z_i}', \overrightarrow{z_{i'}}'\right) \approx 1$, have more chance of being on the top-N recommended list $R.List$ than distant ones i.e. $sim\left(\overrightarrow{z_i}', \overrightarrow{z_{i'}}'\right) \approx 0.5$. We note that 0.5 is considered as the limit of accepted items to be recommended.

4.2.5 Explainable Recommendation

Explain to the user why an item had been recommended allows the RS gaining his confidence and helps the user make good and fast decision, and furthermore preserve the transparency of the RS. So, by adding, an explainable sentence such as ("*You have already liked movies directed by…*", "*you liked this item, you may be interested by this one*", etc.) may be a good justification manner.

In our work, explainable sentences are provided based on the generated set of relation-specific embeddings $\vec{h}_e^{r\prime}$ .Indeed, the explanation sentences are generated based on the computed similarity scores between each vector pair ($\vec{h}^r_{e\in R.List}{}', \vec{h}^r_{e\in u.Profile}{}'$) such as, for each item e in the recommended list $R.List$ we compute the cosine similarity between his each relation-specific vector $\vec{h}^r_{e\in R.List}$ and each relation-specific vector in the target user profile $\vec{h}^r_{e\in u.Profile}{}'$. After that, the results are ranked and the properties that have the highest scores are selected to further used to formulate the explanations. In fact, the relation-specific similarity allows to understand user preferences and then depicts the reason behind the recommendations. We note that the type of the explanations depends on the relations. For example, in the movie domain, we have: starring-based explanations, director-based explanations, genre-based explanation, and finally subject-based explanations. Therefore, we design the following templates for explainable sentences: "*You have already liked movies acted by (Actor_Name)"*, *"You have already liked movies directed by (Director_Name)"*, *"You have already liked (Genre_Label) movies"* and *"You have already liked (Subject_Label) movies"*.

4.3 Experimental Evaluation

In this section, we evaluate our proposed model on two real-world scenarios for both movie and book recommendations.

4.3.1 Experimental Settings

4.3.1.1 Datasets

Two public real-world datasets, MovieLens and LibraryThing, were used to validate the effectiveness of our approach for movie and book recommendations, respectively. The datasets

were randomly split into two sets to perform the experiments. The Train set includes 80% of each user's ratings, and the Test set includes 20%.

- MovieLens. This benchmark dataset contains 943 users, 1,682 movies, and 100,000 explicit ratings on a 5-star scale. Each user has rated at least 20 movies. For the preprocessing phase, we considered both positive and negative ratings. We converted the ratings into binary ratings by reassigning 1 as "relevant" to ratings not lower than 3.0 and 0 as "not-relevant" to all other cases.

- LibraryThing. The second dataset is related to the book domain and contains 7,279 users, 37,231 books, and over 700,000 ratings.

- DBpedia. It is a cross-domain dataset in the Linked Data cloud. This study considered the DBpedia classes *dbo:Film* and *dbo:book*, representing the movie and book domains. We then intuitively selected the most frequent properties used to describe each content, namely *dbo:starring*, *dbo:director*, *dbo:producer*, *dbo:distributor*, *dbo:writer* and *dct:subject* for the movie items and, *dbp:author*, *dbo:publisher, dbp:subjects*, and *dbp:language* for the book items.

4.3.1.2 Baselines

The recommendation results were compared to the following systems, respectively:

-**KGCN** (Wang *et al.*, 2019a). It is the first system to extend GCNs to perform KG-based recommendation.

-**I-KNN**. Item-based K-Nearest Neighborhood is the standard collaborative filtering method algorithm. To predict the rating of item i, KNN first determines K-items similar to the given item. Then, it infers i's rating based on u's ratings on those similar items. The elements closest to i are determined using a similarity metric such as cosine similarity.

-**CB**. Content-based filtering is an important approach to the recommendation. The main idea is to recommend items closest to what the user has previously liked. The content-based RS calculates the similarity between items based on their descriptions or features.

-**MF**. This benchmark factorization model considers second-order feature interactions between the inputs.

4.3.1.3 Metrics

As performance measures for item recommendations, we use both precision and recall metrics. For a user u that receives a list of N recommended items, precision and recall are defined Equation 4.6 and Equation 4.7, respectively:

$$Precision@N(u) = \frac{number\ of\ relevant\ items\ in\ the\ topN\ list}{N} \tag{4.6}$$

$$Recall@N(u) = \frac{number\ of\ relevant\ items\ in\ the\ topN\ list}{total\ number\ of\ relevant\ items} \tag{4.7}$$

4.3.2 Comparative Results

The overall precision and recall value was computed as the average of Precision@N and Recall@N overall users, respectively. Table 4.2 lists the experimental results in terms of Precision@N and Recall@N, with N being 5, 10, 15, and 20 on MovieLens and LibraryThing datasets, respectively.

Table 4.2 The results of *Recall@N* and *Precision@N* in the top-N recommendation.

MovieLens								
	Precision@N				**Recall@N**			
	P@5	***P@10***	***P@15***	***P@20***	***R@5***	***R@10***	***R@15***	***R@20***
Explain-KGCN	0.208	0.277	0.279	0.286	0.215	0.208	0.207	0.199
KGCN	0.196	0.238	0.242	0.248	0.143	0.136	0.128	0.122
I-KNN	0.041	0.066	0.071	0.070	0.034	0.031	0.026	0.021
CB	0.143	0.161	0.168	0.170	0.122	0.122	0.121	0.118
MF	0.110	0.115	0.116	0.117	0.141	0.130	0.125	0.116
LibraryThing								
	P@5	***P@10***	***P@15***	***P@20***	***R@5***	***R@10***	***R@15***	***R@20***
Explain-KGCN	0.108	0.179	0.182	0.184	0.143	0.137	0.122	0.113
KGCN	0.106	0.152	0.158	0.171	0.145	0.142	0.140	0.117
I-KNN	0.032	0.056	0.058	0.061	0.036	0.034	0.028	0.027
CB	0.054	0.136	0.138	0.140	0.058	0.051	0.043	0.039
MF	0.047	0.102	0.109	0.113	0.092	0.081	0.069	0.064

The experiments show that Explain-KGCN globally outperforms the comparative methods and achieves large improvements for both evaluation metrics on both databases. Indeed, using MovieLens base, our method outperformed all other methods (KGCN, I-KNN, CB, and MF). A 5,77% improvement in P@5, 13,74% in P@10, 13,26% in P@15, and 13,29% in P@20 was obtained over the precision achieved with the KGCN method. A 1.38% improvement in R@5, a 3.52% improvement in R@10, a 12.86% improvement in R@15, and a 3.41% improvement in R@20 was respectively obtained over the recall achieved with the KGCN method.

These results are better than those obtained with the I-KNN, CB, and MF methods. The improvements of our system compared with KGCN (the second-best performance) are significant. They are mainly since the proposed algorithm (1) learns from a weighted graph in which entities are weighted according to their importance in the graph. However, KGCN operates on an unweighted knowledge network in which all nodes are treated equally. (2) generates for each item, a global embeddings as well as a set of relation-specific vector representations. These latter allowed to easily identify the semantics encoded in each vector value. Indeed, Explain-KGCN gains explainability using graph-based recommendation as a set of semantically weighted paths.

The proposed KG weighting metric improves the quality of embeddings and thus the prediction performance since it considers three important factors: the KG topology, the semantic relations between the target node and its neighbors, and the negative and positive users' personalized preferences.

The proposed method also outperformed all deep-unaware models. For example, it provided a 237,5% improvement in P@5, 219,6% in P@10, 213,8% in P@15, and 201,6% in P@20 over the precision obtained with the I-KNN method and an 84,5% improvement in R@5, 85,5% in R@10, 87,1% in R@15, and 89,5% in R@20 over the recall obtained with I-KNN. When it comes to the LibraryThing dataset, the performance of all the algorithms decreased since the set of features used to describe a Book item is smaller than those used to describe a Movie item. But our recommendation system performed globally better than the other methods. It achieved a precision of 0.108, 0.179, 0.182 and 0.184 when N = 5, N=10, N = 15, and N = 20 respectively, which is an improvement over KGCN by 1,85%, 6,15%, 12,09%, and 6,52% respectively.

For the recall comparison, the proposed system achieves 0.143, 0.137, 0.122, and 0.113 when N = 5, N = 10, N = 15, and N = 20, without significant improvement over KGCN. This result proves the performance of our system in various domains. In addition, it is worth noting that experimental results on the LibraryThing dataset show the efficiency of our system and its

superiority over such deep-unaware state-of-the-art approaches as I-KNN, MF, and CB in terms of precision and recall. For example, it provided 70.3%, 68,7%, 68,1%, 66,8% improvement in P@5, P@10, P@15, and P@20 respectively over the precision obtained with I-KNN and 74.8% improvement in R@5, 75.2 in R@10, 77% in R@15, and 78% in R@20 over the recall obtained with I-KNN. Moreover, it can be noticed that P@20 and R@20 obtained the best scores, which means that a significant number of items recommended by the system are relevant. This result shows the effectiveness of the proposed approach to better rank relevant items.

4.3.3 Explainability

In this section, we address the research question **RQ1.5** *Is the proposed system can insure the explainability of recommendations?*

Explain to the user why an item had been recommended allows the RS gaining his confidence and helps the user make good and fast decision, and furthermore preserve the transparency of the RS. So, by adding, an explainable sentence such as ("*people who liked this item also liked this one*", "*you liked this item, you may be interested by this one*", etc.) may be a good justification manner.

In our work, explainable sentences are provided based on three aspects that are: the graph topology, the semantic nature of the KG and the personalized aspect. Indeed, the explanation sentences are generated based on the DC_r weights such as, for each recommended item we extract his director, actor and genre. After that, we find in the graph all edges that connect the user with this director, actor, and genre respectively. Then the average DC_r of each set of similar edges is computed and the results are ranked. Finally, the property that has a highest average DC_r is selected and based on it the explanation sentence is formulated.

For example, among movies recommended to user-379 two of them are directed by Steve Oedekerk which are "*la famille folding*" and "*docteur patch*" the explanation sentence were "*you liked most movies directed by Steve Oedekerk*. In fact, the average DC_r attributed to this director was 0.570 comparing to 0.470 attributed to the actor "Jim Carrey" and 0.469 to the comedy genre.

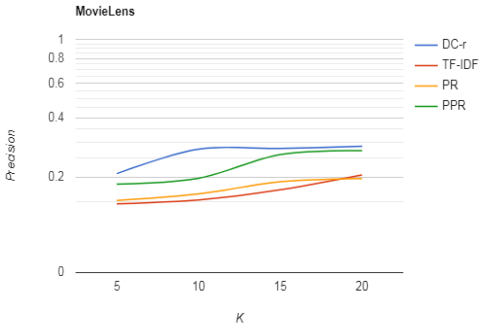
The assessment of the proposed explanations consisted of a simple 5 stars evaluation. The results show that over 169 users 81% of them gave 4 stars at least.

4.3.4 DC_r Impact

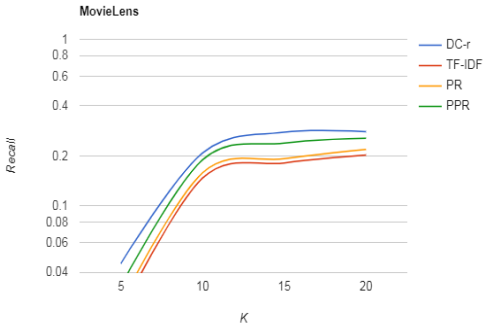
To verify the effectiveness of DC_r, the performance of Explain-KGCN is compared across four different versions:

Explain-KGCN with weights learned by Term frequency-inverse document frequency (TF-IDF), Explain-KGCN with weights learned by PageRank (PR), Explain-KGCN with weights learned by Personalized PageRank (PPR), Explain-KGCN with one-hot encoding (unweighted graph).

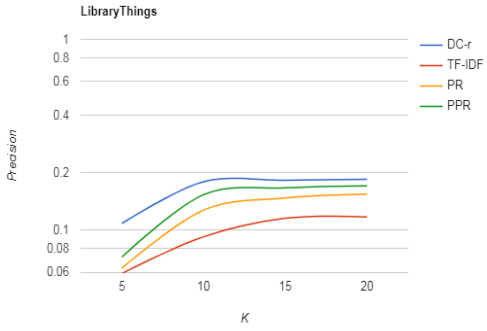
The evaluation metrics used are Precision@K and Recall@K, where K takes values from the set {5, 10, 15, 20}. The average values of Precision@K and Recall@K are displayed in Figures 4.2. Precision@K measures the proportion of correctly recommended items among the top K recommendations. It indicates how many of the recommended items are relevant to the user's preferences. Recall@K, on the other hand, measures the proportion of correctly recommended items out of all the relevant items. It provides an indication of how well the recommendations cover the total set of relevant items.



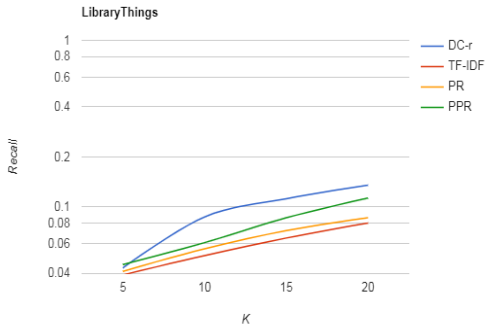
(a)



(c)



(b)



(d)

Figure 4.2. Precision and Recall in MovieLens and LibraryThing datasets, respectively. 4.6 (a) represents the results of Precision@K on MovieLens, 4.2 (b) represents the results of Recall@K on MovieLens, 4.2 (c) represents the results of Precision@K on LibraryThing, and 4.2 (d) represents the results of Recall@K on LibraryThing. The name of each curve is formed with the algorithm name and weighting method name. For example, "Explain-KGCN-PR" means the KGCN algorithm with the PageRank weighting method.

In the MovieLens dataset, Explain-KGCN DC_r consistently achieves the highest precision and recall values across all N values (5, 10, 15, and 20). On the other hand, Explain-KGCN tf-idf demonstrates lower precision and recall values in comparison to Explain-KGCN DC_r, indicating relatively less accurate item recommendations. Meanwhile, Explain-KGCN PR and Explain-KGCN PPR exhibit moderate precision and recall values, with a slight improvement compared to Explain-KGCN tf-idf.

Similar to the MovieLens dataset, in the LibraryThings dataset, Explain-KGCN DC_r outperforms the other methods in terms of precision and recall. Explain-KGCN tf-idf presents lower precision and recall values, suggesting less accurate recommendations. Although Explain-KGCN PR and Explain-KGCN PPR demonstrate higher precision and recall values than Explain-KGCN tf-idf, they still fall short of the performance achieved by Explain-KGCN DC_r.

To summarize, across both datasets, Explain-KGCN DC_r consistently exhibits superior performance compared to the other methods, delivering higher precision and recall values. These results indicate that Explain-KGCN DC_r is more effective in generating accurate recommendations in these particular scenarios.

4.3.5 Case Study

In order to illustrate the effectiveness of Explain-KGCN to offer personalized understandable explanations, we present a real example of MovieLens. we focus on a sampled user *ID. 548*, whose subset profile is {training set: *Saving private Rayan*, *Jupiter ascending*, and *Schindler's list* | test set: *The terminal*, *Good will hunting*, and *Hoop dreams*}. Figure 4.3 gives a visual impression of Explain-KGCN Explainability. First, we use DC_r to weight the different entities. Then, we apply Relation-Specific GCN algorithm in order to generate for each item, its global embeddings $\vec{z_i}'$ as well as the relation-specific embeddings $\vec{h_e^r}'$. For recommendation purpose, we compute the relatedness between the candidate items {*The terminal*, *Good will hunting*, and *Hoop dreams*} and the items previously liked by the user *ID. 548*. The highest cosine-similarity scores are considered for the selection of the list of recommended items: {*The terminal* - *Saving*

private Rayan: 0.89}, {*The terminal - Schindler's list: 0.87*}, {*Good will hunting - Saving private Rayan: 0.75*} and {*Good will hunting - Schindler's list: 0.62*}. So that, the two movies *The terminal* and *Good will hunting* are recommended. To generate explanations, we compute the cosine similarity between each relation-specific vector $\vec{h}^{r}_{e \in R.List}$ in the *R.List* i.e.{*The terminal* and *Good will hunting*} and each relation-specific vector in the target user profile $\vec{h}^{r}_{e \in u.Profile}{}'$ i.e. {*Saving private Rayan*, *Jupiter ascending*, and *Schindler's list*}. For example, the cosine similarity between Starring-embeddings of *The terminal* and Starring-embeddings of *Saving private Rayan* is given as: $sim_{cosine}(\overrightarrow{z^{The\ terminal}_{r_1=starring}}', \overrightarrow{z^{Saving\ private\ Rayane}_{r_1=starring}}')$ =0.8571. So that, we deal with a Starring-based explanation and the sentence: "*You have already liked movies acted by* ***Tom Hanks***" is presented.

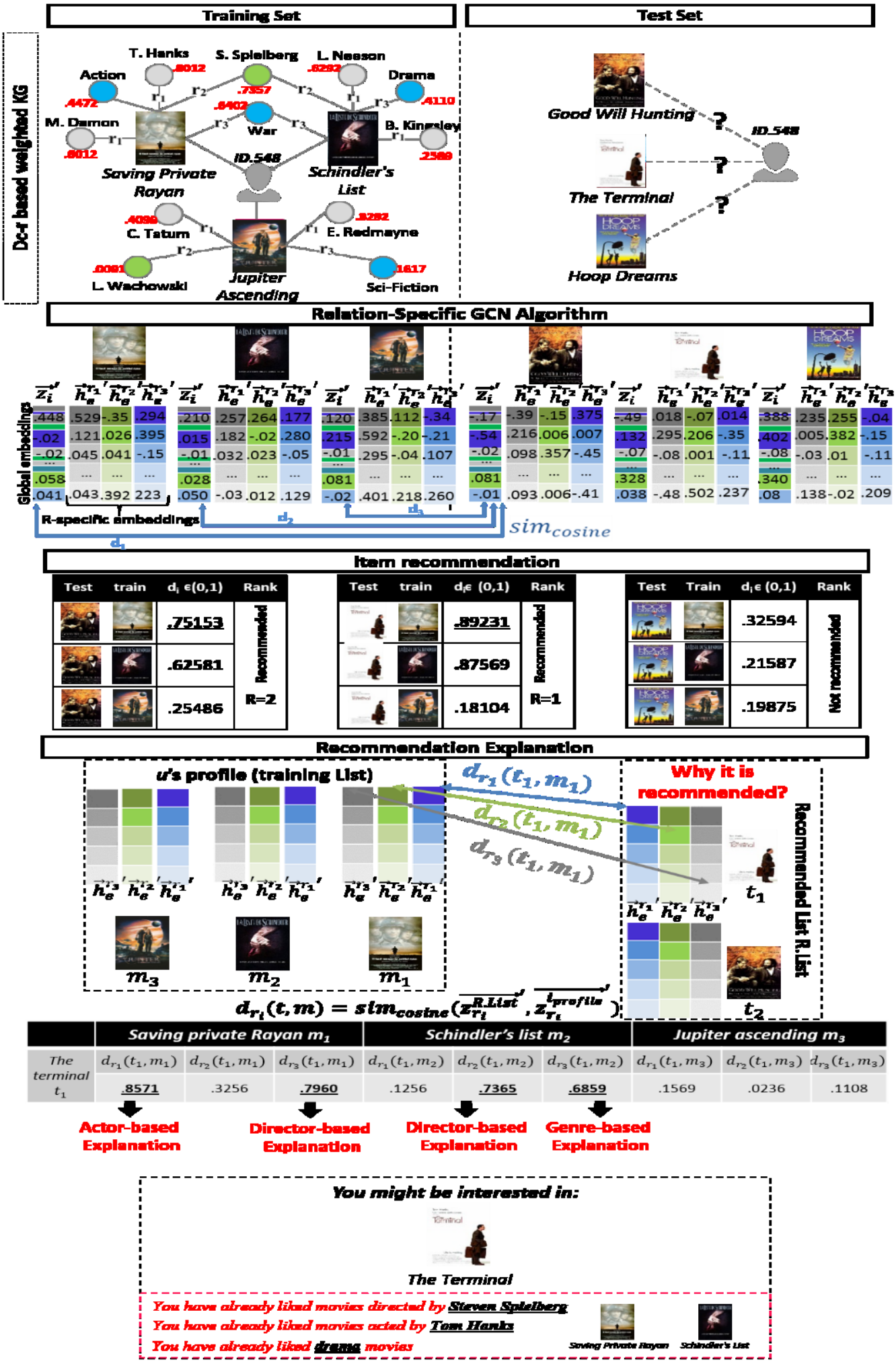


Figure 4.3 Case study of Explain-KGCN process for a sampled user.

Conclusion

In conclusion, this chapter has presented Explain-KGCN, a knowledge-aware Recommender System based on Graph Convolutional Networks. The core component of the proposed system is a relation-specific GCN aggregator function that generates personalized relation-specific vector representations for nodes in the KG by aggregating information from their relation-based neighbors. By explicitly modeling the semantics of various relations, the system enhances the quality of embeddings and prediction performance.

The KG used in this work was constructed by interlinking real-world datasets MovieLens and LibraryThing with the DBpedia knowledge base, resulting in a rich source of background knowledge. Additionally, a novel KG weighting metric DC_r was introduced to measure the importance of each node, taking into account the semantic nature of the KG. The proposed model benefits from considering both explicit positive and negative user feedback to weight different nodes in the graph, enabling a more comprehensive representation of user preferences. For the recommendation process, a relation-specific GCN algorithm was developed to learn vector representations of entities. These representations were then utilized in the similarity computation process to generate a ranked list of Top-N recommended items for each user. Experimental results demonstrated the effectiveness of incorporating background knowledge from the Linked Open Data cloud, resulting in improved recommendation accuracy. Furthermore, incorporating prior information about node importance facilitated model prediction and ensured the full utilization of semantic and collaborative information.

Explain-KGCN demonstrated strong performance in terms of accuracy and provided precise explanations for its recommendations. However, we found the process of weighting to be highly involved and costly, as it required separate tasks for learning embeddings and calculating weights. As a result, the idea emerged to automate the weight calculation process using a tool that is as powerful as GCN but also ensures the weighting process.

Chapter 5

Knowledge Graph Attention for Diverse Explainable Recommendation

Contents

Introduction

As the realm of digital data experiences a rapid expansion, personalized Recommender Systems have evolved into a necessity across a spectrum of domains including e-commerce, social networks, and online media. Over the past few years, Knowledge Graphs (KGs) have emerged as a potent mechanism for depicting and rationalizing complex relationships among entities and attributes within diverse fields.

In the previous chapter, we introduced Explain-KGCN (Boughareb *et al.*, 2023), an innovative knowledge-aware Recommender System built upon the foundation of Graph Convolutional Networks. Diverging from other Recommender Systems within the same research domain that operate on an unweighted Knowledge Graph (KG) input, Explain-KGCN takes a distinctive approach by leveraging a weighted semantic representation. This strategic choice aims to circumvent semantic loss and precisely capture users' preferences. However, the intricacies of the node weighting procedure were notably complex. Furthermore, this approach incurred substantial expenses due to the necessity of separate tasks for embedding acquisition and weight computation. Consequently, the notion emerged to automate the weight calculation procedure through the utilization of a robust tool capable of efficiently streamlining the weight determination process. Graph Attention Networks (GATs), equipped with their inherent attention mechanisms, offer an optimal solution for tackling this challenge. By singling out the most pivotal nodes as an integral facet of the embedding learning process, GATs present an ideal mechanism to overcome this obstacle.

In the current chapter, we embark on a comprehensive examination of the capabilities of Knowledge Graphs (KGs) in realizing a wide array of recommendations that are both versatile and comprehensible. This exploration is achieved through the integration of Graph Attention Networks (GATs). The approach we present capitalizes on the strengths of two preceding approaches: the primary contribution elucidated in Chapter 3 and the secondary contribution expounded upon in Chapter 4. Our approach extends from a comparable Knowledge Graph framework as the initial advancement, wherein we introduce a knowledge representation that fosters the simultaneous acquisition of embeddings for nodes and semantic attributes.

A significant contrast within our proposed methodology lies in the automated property selection process. Instead of intuitively choosing discriminative properties, we employ an

automatic process to identify the most discriminative properties. This enables us to refine the recommendation process further and enhance its interpretability.

Furthermore, our current suggested approach derives inspiration from the concept of weight computation and relation-specific aggregation as seen in Explain-KGCN. However, rather than directly guiding this process, we opt to employ Graph Attention Networks (GATs) to oversee it. GATs offer a robust framework that enables efficient weight computation, seamlessly aligning with our overarching objective of enhancing the recommendation process and ensuring its precision. Through the amalgamation of these components, our proposed approach endeavors to leverage the combined potential of Knowledge Graphs (KGs), Graph Attention Networks (GATs), and automated property selection to realize a spectrum of recommendations that are both diverse and comprehensible.

Within the context of the aforementioned challenges, we explored the above sub-research questions:

- **RQ 1.** *How can a Recommender System harness the effectiveness of the DGRL algorithm GAT to proficiently learn features from graphs?*
- **RQ 2.** *How does the Information Gain (IG) property extraction method contribute to reduce the complexity of GAT?*
- **RQ 3.** *Can the proposed Recommender System guarantee the diversity of the recommended items?*
- **RQ 4.** *Can the proposed Recommender System provide explanations for the diversity observed in its recommendations?*

The remaining content of this chapter is organized as follows. In Section 5.1, we present the main constituents of our approach, starting with the property selection process detailed in Section 5.1.1. This section outlines the automated procedure by which pertinent properties are selected for recommendation. Subsequently, Section 5.1.2 expounds upon the attentive embedding layer implemented within our model, facilitating the capture of comprehensive representations for both entities and properties. Furthermore, in Section 5.1.3, we propose a protocol for diversification, providing insights into how we manage and regulate the diversity of suggested items.

Following the description of our proposed approach, Section 5.2 focuses on the experimental evaluation of our method. In Section 5.2.1, we provide insights into the experimental settings, detailing the datasets and evaluation metrics employed in our experiments. Subsequently, in Section 5.2.2, we present comparative results that highlight the effectiveness of our approach. This section is further divided into three subsections: Section 5.2.2.1 analyzes the impact of property selection on recommendation accuracy, Section 5.2.2.2 compares our approach with baseline methods, and Section 5.2.2.3 delves into the performance of our model in terms of explainable recommendations. Finally, we conclude this chapter, summarizing the key findings and contributions of our work.

5.1. Diversity Explanation Using Graph Attention Networks in Joint Entity-Relation Representation

The proposed approach consists of three main stages. Initially, a subset of properties is selected automatically using the Information Gain method in the first stage, facilitating property selection. Subsequently, KG embeddings are produced using GATs in the second stage, generating entity-specific embeddings that consider the semantic connections within the KG. Finally, the third stage involves providing an explainable ranked list of Top-N items tailored to individual user preferences, referred to as the Top-N explained item recommendation. Further elaborations on each of these stages are presented in the subsequent sections. Figure 5.1 illustrates the architecture of the proposed system, and the following subsections detail each step.

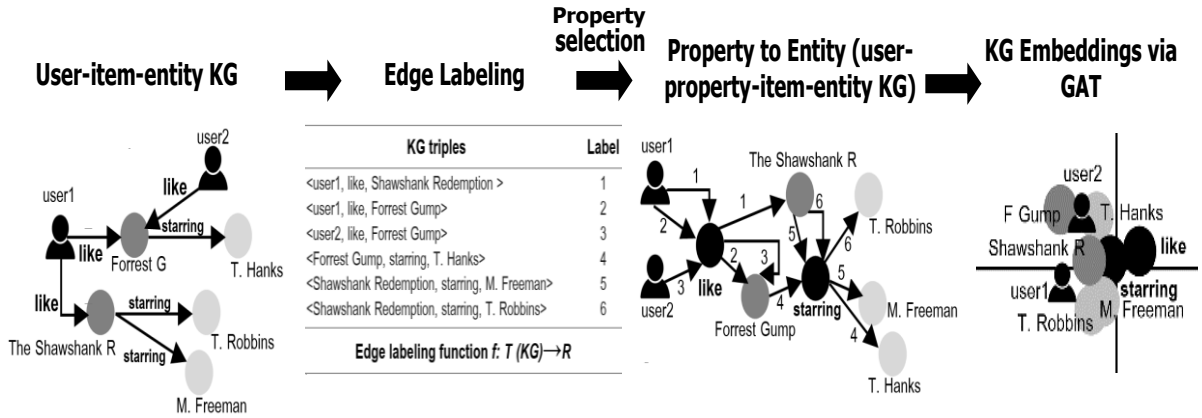


Figure 5.7. The proposed system architecture.

5.1.1 Information Gain for Property Selection

Given that Graph Attention Networks (GATs) can incur significant computational overhead, particularly when utilized alongside Recommendation Systems (RSs) for processing extensive volumes of semantic Linked Data, it becomes imperative to identify the properties that offer the most value in augmenting the efficacy of our recommendation approach and fine-tuning prediction outcomes. To achieve this, we have implemented a technique for property selection, the objective of which is to reduce the number of properties within the dataset.

Property selection serves as a data analysis mechanism that eliminates irrelevant properties from high-dimensional data. This, in turn, contributes to the enhancement of the performance exhibited by machine learning models. Through the judicious selecting of pertinent properties, we can concurrently reduce computational costs and improve the accuracy of our predictions.

Information Gain represents an entropy- based feature selection method, commonly used within the realm of machine learning. Its operational principle lies in quantifying the decrease in entropy or disorder in the target variable by splitting data based on the value of a feature. In a more granular context, this method computes the entropy of the target variable both before and after the division, subsequently computing the disparity between these two entropy values.

The magnitude of this disparity serves as an indicator of the feature's informativeness. Essentially, a greater dissimilarity signifies a higher level of informative content present in the feature. This way, Information Gain adeptly discerns the most pertinent features germane to modeling and predictive undertakings. Consequently, it effectively trims the dimensionality of input data, leading to an enhancement in model efficiency and performance.

The formal definition of Information Gain can be expressed mathematically as follows in Equation 5.1:

$$IG(X,Y) = Entropy(X) - Entropy(X|Y) \qquad (5.1)$$

Where $IG(X,Y)$ represents the information gain acquired by knowing variable Y about variable X, $Entropy(X)$ corresponds to the level of uncertainty present in the probability distribution of variable X. On the other hand, $Entropy(X|Y)$ stands for the conditional entropy of variable X, given the variable Y. The entropy of a random variable X is a measure of the unpredictability in its probability distribution. This measurement involves the summation of the

negative logarithms of the probabilities linked to its potential outcomes, with their respective probabilities as weights. The precise definition of the entropy of variable X is outlined in Equation 5.2.

$$Entropy(X) = -\sum p\,(X)\, log_2\, p(X) \qquad (5.2)$$

In this formula, $p(X)$ denotes the probability associated with the outcome X, the summation encompasses all potential outcomes of X. The unit of measurement for entropy is bits, and it is maximized when all possible outcomes hold equal likelihood, while it decreases to a minimum when a single outcome is certain to materialize with a probability of 1.

In the context of the selection of semantic properties for predictive purposes through information gain, the initial step involves defining a target variable that we want to predict based on the semantic properties. Once this target variable is established, the information gain of each semantic property concerning this target variable can be computed. This calculation allows us to pinpoint semantic properties with the greatest information gain, and these selected properties are then harnessed as features within our machine learning model. In essence, the information gain metric reveals how much information a property contributes to the target variable.

We define the information gain of a property P_i as (Equation 5.3):

$$Gain(P_i) = Entropy(I) - \sum_{v \in p_i} \frac{|I_v|}{|I|} * Entropy(I_v) \qquad (5.3)$$

$$Entropy(P_i) = -\sum_i P(i)\,.\,log_2\, P(i) \qquad (5.4)$$

Where, $Entropy(I_v)$ denoted in Equation 5.4 represents the value of the entropy of the data, wherein $P(i)$ is the probability of obtaining the ith value upon selection from the set. Meanwhile I_v represents how many items have the property P_i with the specific value v, and $Entropy(I_v)$ refers to the entropy computation derived from the data subset in which the property P_i assumes value v. As part of the process, features are systematically arranged in descending order based on their Gain values, and subsequently, the top-k highest-ranked features are earmarked for selection.

5.1.2. Knowledge-Aware Attention-Based Embedding Layer

Graph Attention Networks is a type of neural network architecture designed to operate on graph-structured data. GAT incorporates attention mechanisms, which allows for the adaptive combination of features from neighboring nodes. The core idea behind GAT is to compute the attention coefficients between a node and its neighbors, which determine the importance of each neighbor node's features for the central node. The attention coefficients are computed by applying a neural network to a concatenation of the central node's feature vector and its neighbors' feature vectors. The resulting attention coefficients are then used to compute a weighted sum of the neighbor nodes' feature vectors, which is combined with the central node's feature vector to obtain a refined representation.

We employed the GAT model on the Knowledge Graph $KG\ (E, T)$, where E denotes the set of four types of nodes representing users $U = \{u_1, \ldots, u_n\}$, Items $U = \{i_1, \ldots, i_m\}$, and different semantic entities$s = \{s_1, \ldots, s_k\}$, and semantic properties $p = \{p_1, \ldots, p_l\}$. T denotes the set edges between nodes. The following pseudo-algorithm outlines the key steps involved in running the GAT model on a Knowledge Graph KG and updating node embeddings using attention mechanism.

Algorithm 1 Attentive Knowledge Graph Embedding

Input: Node embeddings e_i

Output: Updated node embeddings $\overrightarrow{z_{e_i}}$

//1. Initialize node embeddings:

for i in $\{1, 2, \ldots, N\}$

$e_i = \{0, 1\}$

// N is the total number of nodes.

//2. Repeat for each node e_i in KG:

//a. Compute attention scores for node e_i and its neighbors e_j:

$$\alpha(e_i, e_j) = softmax\left(\text{LeakyReLU}\left(a^T \cdot \left[w_{e_i} | w_{e_j}\right]\right)\right)$$

//b. Calculate the weighted sum of neighbor embeddings for node e_i:

$$h_{e_i} = \sigma\left(\Sigma_v \in e_i\ \alpha(e_i, e_j) w_{e_j}\right)$$

//c. Update the output embedding for node e_i:

$$\overrightarrow{z_{e_i}} = \overrightarrow{z_{e_i}} \parallel h_{e_i}$$

Our objective is to generate vector representations for each input node, denoted as $e_i = \{e_1, e_2, ..., e_N\}$, where e_i is the input embedding of the i-th node, and $\{\overrightarrow{z_{e_i}} = \overrightarrow{z_{e_1}}, \overrightarrow{z_{e_i}}, ..., \overrightarrow{z_{e_N}}\}$ represents the output embeddings of these nodes. N signifies the total node count in the context. The attention score α of a node e_i is given in Equation 5.5 (Step 2.a) :

$$\alpha(e_i, e_j) = softmax\left(\sigma\left(a^T \cdot \left[w_{e_i} | w_{e_j}\right]\right)\right) \qquad (5.5)$$

where W represents a learnable linear projection matrix, σ represents *LeakyReLU* activation function, it gives the final functional form for calculating raw attention scores which we can then pass into a *softmax* function to get the final normalized attention scores α. a is a learnable parameter *that* represents the direction that should be attended to. Indeed, the equation applies a linear projection to each of the embedding separately, concatenates the results, and takes the dot product with a (Step 2.b). The final form of the GAT update equation can be defined as (Equation 5.6) (Step 2.c):

$$h_{e_i} = \sigma\left(\Sigma_v \in e_i \, \alpha(e_i, e_j) w_{e_j}\right) \qquad (5.6)$$

5.1.3. Diverse Item Recommendations

The item recommendation problem involves providing a ranked list of items that are likely to interest a user. Similar to our first contribution (Chapter 3), we employ the score function derived from the TransE embedding method (Bordes *et al.*, 2013). This scoring method has shown adaptability to KGs, surpassing cosine similarity measures (Chapter 3, Section 3.6). TransE is specifically designed to learn representations of entities and relations in knowledge graphs, ensuring that the equation s + p ≈ o holds true for an RDF triple <s, p, o>.

To tackle the challenge of enhancing diversity within our RS, we introduce a well-defined re-ranking protocol. This protocol involves the identification of the minimum relatedness score between items and properties. The fundamental concept behind this approach is to gauge an item's least pronounced association with semantic relations, signifying its distinctiveness from the user's favored properties. This protocol leverages property embeddings to quantify these relationships effectively.

In the ranking process, we first evaluate each item beyond the k-th position in the recommended item list (RI-list) by calculating its relatedness score with respect to each property. Then, among all the properties, we identify the minimum relatedness score for each item, which

subsequently becomes its diversity score. Utilizing these diversity scores, we proceed to rank the items beyond position k in descending order, resulting in a thoughtfully curated list that offers users a more diverse set of recommendations tailored to their preferences and interactions. Algorithm 2 represents the pseudocode for the diversity scoring and ranking process:

Algorithm 2 Property-based diversification

Input:

RI_list: Recommended item list

k: Position threshold for top-N recommendations

p_list: List of properties

Output: $sorted\_items$

Initialize an empty dictionary to store diversity scores for items

$diversity\_scores = \{\}$

for $item$ in $RI\_list[k:]$:

  $relatedness\_scores = []$

  for $property$ in p_list:

    $relatedness\_score = d(item, property)$

    $diversity\_scores[item] = min\_relatedness\_score$

    # Sort items beyond position k based on their diversity scores in descending order

    $sorted\_items = sorted(diversity\_scores.keys(), key = lambda\ x: diversity\_scores[x])$

Output the sorted list of items beyond position k based on diversity scores

  return $sorted\_items$

This method highlights items that are the least related to any property, ensuring a stronger diversification from the user's preferences. Finding the minimum relatedness score prioritizes items that are distinctly different from the user's preferred properties.

5.1.4. Property-based Diversity Explanations

In this section, we aim to challenge the narrow perception of explainability, which often exists as an isolated metric independent of other recommendation metrics. Our approach involves addressing two fundamental objectives simultaneously: ensuring diversity within recommendations and elucidating the essence of this diversity. This approach entails computing

similarity scores between each recommended item and the user's profile, using embedding vectors. Based on these scores, we identify the most relevant attributes contributing to the recommendations. This empowers us to generate explanatory explanations that shed light on both the user's preferences and the rationale behind the suggested items. We propose five different property-based diversity explanation templates that provide both a reason for the recommendation and an indication of its diversity:

- We recommend [Item_Name] because it aligns with your interests in [Property_Label], much like [Item_Name]. What makes this recommendation unique is its distinct [Property_Type], offering you a refreshing change from your usual choices.

- Our recommendation of [Item_Name] is influenced by its relevance to [Property_Label], similar to [Item_Name]. This choice stands out due to its diverse attributes, adding variety to your selections.

- We've selected [Item_Name] for you, considering your affinity for [Property_Label], similar to [Item_Name]. This recommendation offers a unique twist, as it diverges from your typical choices. We believe it's worth exploring for its distinctive [Property_Type].

- We think you'll appreciate [Item_Name] based on your interest in [Property_Label], much like [Item_Name]. What sets this recommendation apart is its fresh perspective and unique [Property_Type]. Explore new horizons while staying true to your preferences.

- We've chosen [Item_Name] because it resonates with your preferences in [Property_Label], much like [Item_Name]. Add a touch of diversity to your selections, offering a change of pace with its distinctive [Property_Type].

These templates aim to explain why a particular item is recommended to the user while emphasizing its uniqueness or diversity in comparison to their usual choices.

5.2 Experimental Evaluation

In this section, we assess our proposed model in three real-world scenarios for movie, book, and music recommendations. Our objective is to address the following research questions:

- ***RQ1.*** *Can selecting specific properties result in an increase in recommendation accuracy?*
- ***RQ2.*** *How can a Recommender System leverage the efficacity of the DGRL algorithm GAT in learning features from graphs?*
- ***RQ3.*** *Can the proposed Recommender System ensure the diversity of recommendations?*
- ***RQ4.*** *Can the proposed Recommender System provide an explanation for the diversity of recommendations?*

5.2.1 Experimental Settings

5.2.1.1 Datasets. Three benchmark datasets have been selected as the experimental datasets, and two linked datasets have been chosen to provide metadata to the system, **(1)** *MovieLens,* which is a movie dataset, it contains 943 users, 1,682 movies, and 100,000 explicit ratings on a 5-star scale. Each user has rated at least 20 movies; **(2)** *LibraryThing*, which is related to the book domain and contains 7,279 users, 37,231 books, and over 700,000 ratings; **(3)** *Last.fm*, which is a popular music dataset that contains approximately 1.1 billion play counts from 2.1 million users. The dataset includes metadata on artists and tracks, as well as user profiles and listening histories; **(4)** *DBpedia* (DBP in short), which is a cross-domain dataset in the Linked Data cloud; and **(5)** *LinkedMDB* (LMDB in short), which is an open semantic database for movies.

5.2.1.2 Baselines. To evaluate the recommendation results, we compared them to several established baselines, including: Item-based K-Nearest Neighbors (I-KNN), Content-Based Filtering (CB), Matrix Factorization (MF), KG-based Node2vec approach, as presented in our first contribution in Chapter 3 (Boughareb *et al.*, 2019b), Explain-KGCN, as introduced in our second contribution in Chapter 4 (Boughareb *et al.*, 2023)

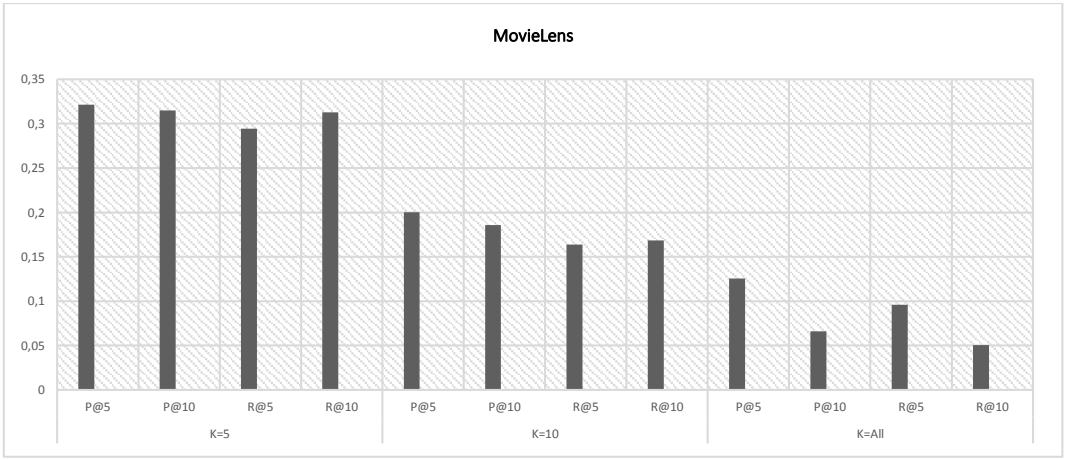
5.2.1.3 Metrics. In evaluating the performance of our item recommendation system, we employ precision, recall, and diversity metrics as key indicators. Specifically, to quantify precision, recall, and diversity for a user u who receives a list of N recommended items, we utilize the following equations: Equation 4.6 (Chapter 4) for precision, Equation 4.7 (Chapter 4) for recall, and Equation 3.4 (Chapter 3) for diversity.

5.2.2 Comparative Results

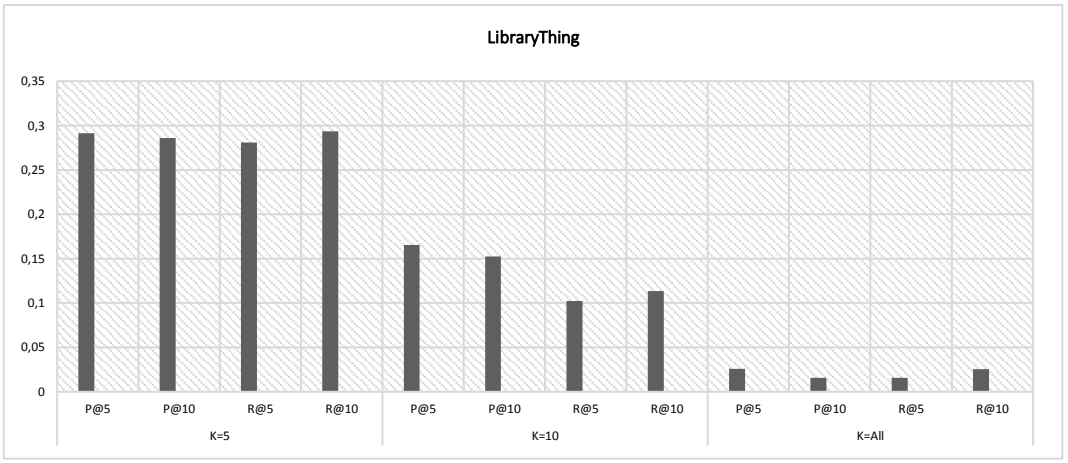
5.2.2.1 Property Selection Impact for Accuracy

In this section, we respond to **RQ1.** *Can selecting specific properties result in an increase in recommendation accuracy?* To measure the impact of property selection on accuracy, we evaluate the performance of the proposed system based on the number of properties (K) used in the learning process, where K is set to 5, 10, and all (All means using all available properties). Our evaluation metrics are precision and recall, which are computed as the average of P@N and R@N across all users. The results of P@N and R@N with N=5 and 10 on the MovieLens, LibraryThing, and Last.fm datasets are presented in Figure 5.1.

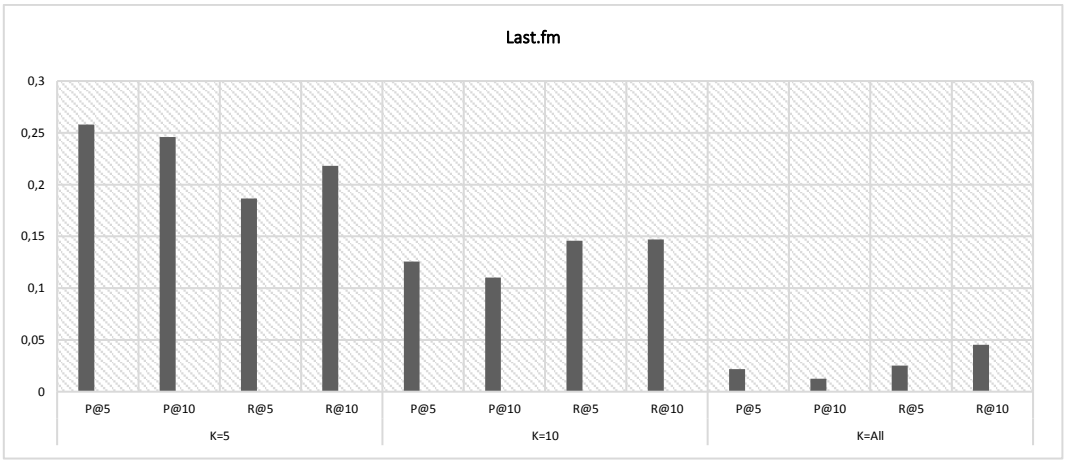
The results show that in the MovieLens domain, the proposed approach achieves the best performance when only 5 properties are used (K=5). However, as the number of features used in training increases, the performance of the model decreases. Similarly, in the LibraryThing and Last.fm datasets, the proposed approach achieves the best performance when K=5, which suggests that a smaller number of properties may be more effective in these domains.



(a)



(b)



(c)

Figure 5.2 Performance of the proposed according to the number of properties (K) used in the learning process. The three datasets (MovieLens, LibraryThing, and Last.fm) are evaluated using precision at N (P@N) and recall at N (R@N) with N set to 5 and 10. The available properties are split into three groups: K=5, K=10, and using all available properties (All). The results indicate that the proposed approach achieves the best performance when only 5 properties are used (K=5) for all three datasets.

5.2.2.2 Comparison with Baselines

In this section, we address **RQ2.**, which examines the performance of the proposed approach and baselines, and **RQ3.**, which investigates the impact of knowledge graphs on the diversity of recommendations. Specifically, we compare proposed system variant with a parameter value of K=5, which achieved the highest precision and recall scores in the previous test, against CB, I-KNN, Explain KGCN, and MF baselines. To assess the system's performance, we use precision, recall, and diversity metrics, which are computed as the average of P@N, R@N, and ILD@N across all users. Tables 5.1(a), 5.1(b), and 5.1(c) present the experimental results for

P@N, R@N, and ILD@N with N=5 and 10 on the MovieLens, LibraryThing, and Last.fm datasets.

We can see from Table 5.1(a), Table 5.1(b), and Table 5.1(c) that the proposed approach outperforms competing systems for all datasets in terms of precision, recall, and diversity. Based on the Table 5.1(a) of the movielens dataset, it appears that the proposed approach achieves P@5 of 0.3214 and P@10 of 0.3150, while the other methods have much lower precision scores, with I-KNN being the lowest. In terms of recall at 5 and 10 recommendations (R@5 and R@10), the proposed approach also performs relatively well, achieving R@5 of 0.2944 and R@10 of 0.3127. Again, the other methods have much lower recall scores, with I-KNN being the lowest. When it comes to the diversity of recommendations, measured by the intra-list diversity (ILD), the proposed approach still outperforms the other methods, but the gap is not as significant as in precision and recall. The proposed approach achieves ILD@5 of 0.2623 and ILD@10 of 0.2702, while the other methods have slightly lower ILD scores.

Similarly, for the LibraryThing dataset (See Table 5.1(b)), the proposed approach's first variant exhibited the highest performance across all evaluation metrics. It obtained a P@5 of 0.2581 and P@10 of 0.2459, followed by CB with a P@5 of 0.1562 and P@10 of 0.1353. Interestingly, the proposed approach performed less effectively in the ILD evaluation metric than in the Recall and Precision metrics. Explain-KGCN performed better than the previous methods but still fell behind the proposed approach in terms of precision. Furthermore, the proposed approach also excelled in terms of recall. It achieved an R@5 of 0.2810 and an R@10 of 0.2936, signifying its capability to retrieve a substantial number of relevant items within the top 5 and top 10 recommendations.

Regarding the Last.fm dataset, the proposed approach performed well compared to the other methods but showed a lower performance than in the other datasets. it achieved a P@5 of 0.2581 and P@10 of 0.2459, followed by CB with a P@5 of 0.1562 and P@10 of 0.1353. Notably, the Item-KNN (I-KNN) method showed a relatively high performance in the Recall evaluation metric (R@5 and R@10) but was significantly less effective in the Precision and ILD metrics. Explain-KGCN surpasses both CB and MF. However, it didn't reach the diversity levels of the proposed approach.

Table 5.1. Performance results for P@5, P@10, R@5, R@10, ILD@5, and ILD@10 on three datasets Movielens, Last.fm, and LibraryThing. The results show that the first variant of the proposed approach performs well on all datasets, with particularly impressive outcomes for the MovieLens dataset.

	P@5	***P@10***	***R@5***	***R@10***	***ILD@5***	***ILD@10***
Proposed approach V1	0,3214	0,3150	0,2944	0,3127	0,2623	0,2702
I-KNN	0,0763	0,0410	0,1569	0,1745	0,0045	0,0112
CB	0,1691	0,1433	0,1254	0,1405	0,0182	0,0633
MF	0,1164	0,1302	0,0261	0,1158	0,0216	0,0251
Explain-KGCN	0,2082	0,2774	0,2157	0,2083	0,0022	0,0152

(a) MovieLens

	P@5	***P@10***	***R@5***	***R@10***	***ILD@5***	***ILD@10***
Proposed approach V1	0,2912	0,2858	0,2810	0,2936	0,1691	0,2026
I-KNN	0,0323	0,0569	0,0612	0,0363	0,0123	0,0258
CB	0,0541	0,1360	0,1405	0,0586	0,0012	0,0023
MF	0,0472	0,1027	0,1137	0,0920	0,0140	0,0146
Explain-KGCN	0,1079	0,1790	0,1433	0,1372	0,0102	0,0058

(b) LibraryThing

	P@5	***P@10***	***R@5***	***R@10***	***ILD@5***	***ILD@10***
Proposed approach V1	0,2581	0,2459	0.1865	0,2180	0,1759	0,1791
I-KNN	0,0182	0,0177	0,1235	0,1244	0,0023	0,0036
CB	0,1562	0,1353	0,0658	0,0431	0,0014	0,0011
MF	0,0211	0,0180	0,0215	0,0324	0,0024	0,0074
Explain-KGCN	0,1235	0,1388	0,1124	0,1154	0,0026	0,0064

(c) Last.fm

5.2.2.3. Explain the Diversity of Recommendations (Case Study)

In this section, we respond to **RQ4.** *Can the proposed Recommender System provide an explanation for the diversity of recommendations?*

To illustrate the effectiveness of the approach to offer personalized understandable explanations, we present a real example of the dataset. we focus on a sampled user ID. 1236, whose subset profile is {itemId: 101 = "Secrets of the Past", itemId: 102 = "The Enigmatic Detective", itemId: 103 = "The Time Traveler's Dilemma", itemId: 104 = "A Twist in Time",

itemId: 105 = "The Labyrinth of Clues", itemId: 106 = "Puzzle of Shadows", itemId: 107 = "Whispers in the Attic", itemId: 108 = "The Silent Witness", itemId: 109 = "Chronicles of a Bygone Era", itemId: 110 = "A World Unseen", itemId: 111 = "The Literary Explorer", itemId: 112 = "Beyond the Horizon}.

Based on the user profile outlined below, we offer personalized book recommendations with explanations using the proposed templates (See Figure 5.3):

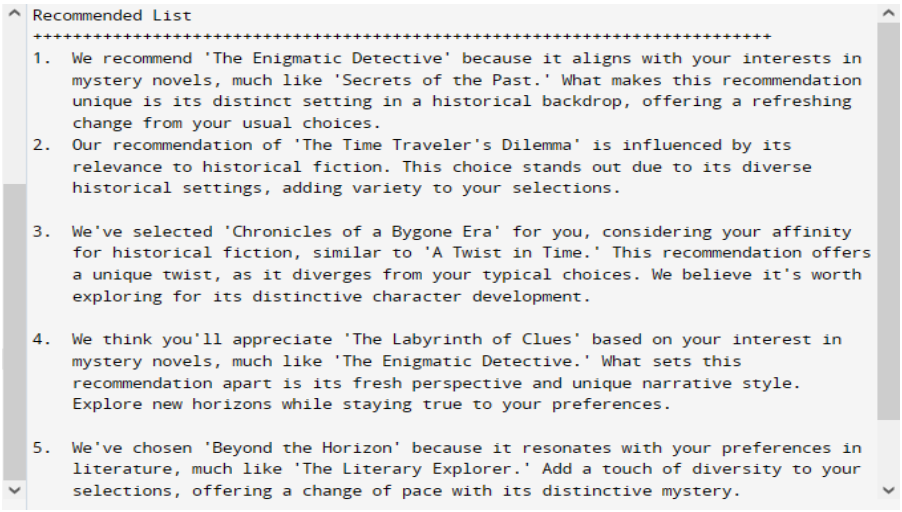


Figure 5.3. The interface of diversity explanations for recommendations for UserID 1236.

- We recommend 'The Enigmatic Detective' because it aligns with your interests in mystery novels, much like 'Secrets of the Past.' What makes this recommendation unique is its distinct setting in a 'historical backdrop', offering a refreshing change from your usual choices.

- Our recommendation of 'The Time Traveler's Dilemma' is influenced by its relevance to historical fiction. This choice stands out due to its diverse 'historical settings', adding variety to your selections.

- We've selected 'Chronicles of a Bygone Era' for you, considering your affinity for historical fiction, similar to 'A Twist in Time'. This recommendation offers a unique twist, as it diverges from your typical choices. We believe it's worth exploring for its distinctive 'character development'.

- We think you'll appreciate 'The Labyrinth of Clues' based on your interest in mystery novels, much like 'The Enigmatic Detective.' What sets this recommendation apart is its fresh perspective and unique narrative style. Explore new horizons while staying true to your preferences.

- We've chosen 'Beyond the Horizon' because it resonates with your preferences in literature, much like 'The Literary Explorer.' Add a touch of diversity to your selections, offering a change of pace with its distinctive mystery.

Conclusion

This chapter proposes to investigate the combination of KGs and GATs for personalized explainable and diverse recommendation, with a focus on addressing the semantic loss problem. Through the proposed approaches and contributions, we have demonstrated the efficacy of leveraging GATs in joint entity-relation representation for semantically-preserving KG-based RS. The inclusion of a semantic-based Information Gain property selection phase has successfully reduced the complexity of the GAT learning process, leading to improved performance. The utilization of the Deep Representation Learning algorithm GAT has allowed us to learn item embeddings and generate diverse and explainable recommendations. These findings contribute to the broader goal of enhancing the user experience and satisfaction with recommender systems.

Conclusion and Future Work

In this thesis, novel methods for utilizing Knowledge Graphs (KGs) in Recommender Systems (RSs) are explored and suggested. With a focus on semantic preservation, this work makes use of developments in machine learning to address problems at the nexus of semantics and RS. In order to fully utilize KGs and effectively reflect user preferences, the thesis underlines the importance of effective knowledge presentation.

Test cases from many domains, including movies, literature, and music, were chosen in order to assess the suggested KG-based recommendation systems. The evaluation considers multiple dimensions, including accuracy, diversity, and explainability.

Three major contributions are presented in the thesis.

First, a Knowledge Representation is proposed that aligns KGs with the user-item bipartite graph for recommendation purposes. Predicates are converted into entities to preserve semantics during the learning phase. This ensured that the Graph Representation Learning algorithm Node2vec could learn embeddings for both entities and properties, which were previously overlooked despite their importance. The proposed knowledge design enables better understanding of user preferences and addresses the diversity problem. The results in the movie domain show promising performance in terms of precision and diversity. This contribution highlights the need for exploring effective data representation techniques that leverage KGs for deep user preference modeling.

Second, the thesis introduces Explain-KGCN, an Explainable KG-aware RS. It incorporates a weighted Knowledge Representation based on a novel relational weighting metric for KGs. The metric assigns personalized weights to nodes, considering prior information about their importance. The algorithm utilizes Graph Convolutional Networks (GCNs) to generate low-dimensional representations and offers effective explanations. Experimental results demonstrate improved performance compared to state-of-the-art approaches.

Building upon the second contribution, the third contribution automates the weighting process using Graph Attention Networks (GATs), a commonly-used architecture in Graph Neural Networks. GATs learn attention coefficients for each node, determining the importance of neighbors in the representation learning process. This contribution explores the potential of KGs for interpretable recommendation using GATs and incorporates additional knowledge from the LOD cloud. The proposed approach improves predictive performance and leverages the rich structured information of KGs to provide explanations for recommendations.

Collectively, these contributions advance the field of KG-based RS by emphasizing the importance of effective knowledge representation and the contributions of KGs in achieving improved accuracy, diversity, and explainability in personalized recommendation.

Future research

The findings of this thesis open up new avenues for further exploration, encouraging researchers to explore the untapped potential of KGs and their applications in various domains. In the field of RS, one important avenue for future research is cross-domain recommendation based on KGs. This involves recommending items from different domains based on user interests. For example, a recommender system could suggest a movie based on a recommended book, or recommend music based on a movie the user likes. This can provide users with a richer and more engaging experience. Another area of future work in the domain of temporal KG-based recommendation. Temporal KGs capture time-dependent relationships and dynamics, allowing RSs to make time-aware recommendations. Investigating the use of temporal KGs in RSs can enable the delivery of more contextually relevant and timely recommendations, reflecting users' changing preferences and interests over time.

Furthermore, considering that this thesis primarily focused on content-based filtering, it is worth exploring hybrid recommendation systems with KGs and advances Deep Graph Learning can be further improve the overall recommendation quality.

In addition to the recommendation domain, it is worth discussing the potential applications of knowledge graphs (KGs) in other domains. KGs have the potential to be powerful tools in various domains, such as healthcare, finance, or education. Investigating how KGs can be effectively utilized in these domains can lead to significant advancements and improvements.

For instance, in healthcare, KGs can help in personalized treatment recommendations, disease diagnosis, and knowledge sharing among medical practitioners.

Bibliography

Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., & Smola, A. J. (2013). Distributed large-scale natural graph factorization. In Proceedings of the 22nd International Conference of World Wide Web (WWW) (pp. 37–48). Rio de Janeiro, Brazil: ACM. doi: 10.1145/2488388.2488393.

Ahmed, M., Anjomshoaa, A., Asfandeyar, M., Tjoa, A. M., & Khan, A. (2010). Towards an ontology-based solution for managing license agreement using semantic desktop. In 2010 International Conference on Availability, Reliability and Security (pp. 309-314). Krakow, Poland. doi: 10.1109/ares.2010.104.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). DBpedia: A nucleus for a Web of open data. In Proceedings of the 6th International The Semantic Web Conference (ISWC) (pp. 722-735). Springer-Verlag. doi= http://dx.doi.org/10.1007/978-3-540-76298-0_52.

Ayundhita, M. S., Baizal, Z. K. A., & Sibaroni, Y. (2019). Ontology-based conversational recommender system for recommending a laptop. Journal of Physics: Conference Series, 1192.

Bag, S., Kumar, S., Awasthi, A., & Tiwari, M. K. (2019). A noise correction-based approach to support a recommender system in a highly sparse rating environment. Decision Support Systems, 118, 46–57.

Baizal, Z. K. A., Iskandar, A., & Nasution, E. (2016). Ontology-based recommendation involving consumer product reviews. 2016 4th International Conference on Information and Communication Technology (ICoICT), 1-6. doi: 10.1109/ICoICT.2016.7571890.

Bechhofer, S., Van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., & Stein, L. A. (2004). OWL web ontology language reference. W3C recommendation, 10(2), 1-53.

Berners-Lee, T. (2000). Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor. New York: HarperCollins Publishers.

Berners-Lee, T. (2006). Linked Data. http://www.w3.org/DesignIssues/LinkedData.html.

Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F., & Secret, A. (1994). The world-wide web. Communications of the ACM, 37(8), 76-82.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. Scientific American, 284(5), 34-43.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008, June). Freebase: a collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD '08), Vancouver, Canada, June 9-12, (pp. 1247-1250). ACM. https://doi.org/10.1145/1376616.1376746

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In Proceedings of the 26th Conference on Neural Information Processing Systems (pp. 2787-2795). NIPS '13. ACM, New York, NY, USA.

Boughareb, D., Khobizi, A., Boughareb, R., Farah, N., & Seridi, H. (2019a). Tag recommendation for limited access papers annotation. In Proceedings of the 8th International conference on innovation and new trends in information technology (INTIS'19) (pp. 245-251). Tangier, Morocco.

Boughareb, D., Khobizi, A., Boughareb, R., Farah, N., & Seridi, H. (2020). A graph-based tag recommendation for just abstracted scientific articles tagging. International Journal of Cooperative Information Systems, 29(3). doi: 10.1142/S0218843020500045.

Boughareb, R., Seridi, H., & Beldjoudi, S. (2019b). Knowledge graph embeddings via node2vec for movie recommendation. In Proceedings of the 8th International Conference on Innovation and New Trends in Information Technology (INTIS) (pp. 27-31). Tangier, Morocco.

Boughareb, R., Seridi, H., & Beldjoudi, S. (2023). Explainable recommendation based on weighted knowledge graphs and graph convolutional networks. Journal of Information and Knowledge Management, 22(3). doi: 10.1142/S0219649222500988.

Boughareb, R., Seridi, H., & Beldjoudi, S. (Forthcoming). Improving explainability and diversity in recommender systems with the use of graph attention networks in joint entity-relation representation. Journal of Web Engineering.

Brickley, D., & Guha, R. V. (2004). RDF vocabulary description language 1.0: RDF Schema. W3C Recommendation.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 12(4), 331-370.

Cai, X., Zheng, Y., Yang, L., Dai, T., & Guo, L. (2018). Bibliographic network representation based personalized citation recommendation. IEEE Access, 7, 457–467. doi: 10.1109/ACCESS.2018.2885507.

Cantador, I., Fernández, M., Vallet, D., Castells, P., Picault, J., & Ribière, M. (2008). A multi-purpose ontology-based approach for personalized content filtering and retrieval. In Advances in Semantic Media Adaptation and Personalization (pp. 25–51). Springer.

Cao, S, W Lu and Q Xu (2015). Grarep: learning graph representations with global structural information. In Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM), pp. 891–900. Melbourne, Australia, ACM, https://doi.org/10.1145/2806416.2806512.

Cao, Y., Wang, X., He, X., Hu, Z., & Chua, T.-S. (2019). Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In Proceedings of the World Wide Web Conference (pp. 151-161).Belkin and Niyogi, 2002

Chen, C., Cui, J., Liu, G., Wu, J., & Wang, L. (2020). Survey and open problems in privacy preserving knowledge graph: Merging, query, representation, completion and applications. arXiv preprint arXiv:2011.10180.

Chen, J, W Yangyang, F Lu, L Xiang, Z Haibin, Y Shanqing and X Qi (2019). N2VSCDNNR: A local recommender system based on node2vec and rich information network. IEEE Transactions on Computational Social Systems, 6(3), 456–466, doi: 10.1109/TCSS.2019.2906181.

Chen, J., Yangyang, W., Lu, F., Xiang, L., Haibin, Z., Shanqing, Y., & Qi, X. (2019). N2VSCDNNR: A local recommender system based on node2vec and rich information network. IEEE Transactions on Computational Social Systems, 6(3), 456-466. doi: 10.1109/TCSS.2019.2906181.

Chicaiza, J., & Díaz, P. V. (2021). A comprehensive survey of knowledge graph-based recommender systems: technologies, development, and contributions. Information, 12, 232. doi:10.3390/info12060232.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12(Aug), 2493-2537.

Dai, Q., Wu, X. M., Fan, L., Li, Q., Liu, H., Zhang, X., Wang, D., Lin, G., & Yang, K. (2022). Personalized knowledge-aware recommendation with collaborative and attentive graph convolutional networks. Pattern Recognition, 128. doi: 10.1016/j.patcog.2022.108628.

Deshpande, M., & Karypis, G. (2004). Item-based top-N recommendation algorithms. ACM Transactions on Information Systems (TOIS), 22(1), 143-177.

Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In Recommender Systems Handbook (pp. 107-144). Springer.

Di Noia, T., Mirizzi, R., Ostuni, V., & Romito, D. (2012a). Exploiting the web of data in model-based recommender systems. In Proceedings of the 6th ACM Conference on Recommender Systems (RecSys '12), September 9–13, Dublin, Ireland, ACM

Di Noia, T., Mirizzi, R., Ostuni, V., Romito, D., & Zanker, M. (2012b). Linked open data to support content-based recommender systems. In Proceedings of the 8th International Conference on Semantic Systems, Graz, Austria, 5–7 September 2012; pp. 1–8, ACM, New York, NY, USA.

Di Noia, T., Ostuni, V., Rosati, J., Tomeo, P., Di Sciascio, E., Mirizzi, R., & Bartolini, C. (2016). Building a relatedness graph from Linked Open Data: A case study in the IT domain. Expert Systems with Applications, 44, 354-366.

Ehrlinger, L., & Wöß, W. (2016). Towards a definition of knowledge graphs. In SEMANTICS 2016: Posters and Demos Track, September 13-14, 2016, Leipzig, Germany.

Fayyaz, Z., Ebrahimian, M., Nawara, D., Ibrahim, A., & Kashef, R. (2020). Recommendation systems: Algorithms, challenges, metrics, and business opportunities. Applied Sciences, 10, 7748.

Fensel, D., Kiryakov, A., Simov, K., Simeonov, K., & Bontcheva, K. (2020). Introduction: What is a knowledge graph?. In Knowledge Graphs (pp. 1-20). Springer, Cham. doi: 10.1007/978-3-030-37439-6_1.

Gardner, H. (2005). Multiple intelligences after twenty years. American Educational Research Journal, 42(4), 881-932.

Ge, M., Delgado-Battenfeld, C., & Jannach, D. (2010). Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In Proceedings of the 4th ACM Conference on Recommender Systems (RecSys), Barcelona, Spain, September 26-30 (pp. 257-260).George, G., & Lal, A. M. (2019). Review of ontology-based recommender systems in e-learning. Computers & Education, 142, 103642.

Ghauth, K. I., & Abdullah, N. A. (2010). Learning materials recommendation using good learners' ratings and content-based filtering. Educational Technology Research and Development, 58, 711–727.

Goble, C., & Stevens, R. (2008). State of the nation in data integration for bioinformatics. Journal of Biomedical Informatics, 41(5), 687-693.

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. Communications of the ACM, 35, 61–70.

Grover, A., & Leskovec, J. (2016). Node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (pp. 855-864). ACM.

Guia, M., Silva, R. R., & Bernardino, J. (2019). A hybrid ontology-based recommendation system in e-commerce. Algorithms, 12, 239.

Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., & He, Q. (2022). A survey on knowledge graph-based recommender systems. IEEE Transactions on Knowledge and Data Engineering, 34(8).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735–1780. doi: 10.1162/neco.1997.9.8.1735.

Hogan, A., Harth, A., Umbrich, J., Kinsella, S., Polleres, A., & Decker, S. (2021). Knowledge graphs. ACM Computing Surveys, 54(4). doi: 10.1145/3447772.

Islam, A., Mohammad, M. M., Sarathi, D. S., & Ali, M. E. (2022). A survey on deep learning based Point-of-Interest (POI) recommendations. Neurocomputing, 306–325. https://doi.org/10.1016/j.neucom.2021.05.114.

Iwahama, K., Hijikata, Y., & Nishida, S. (2004). Content-based filtering system for music data. In Proceedings of the 2004 International Symposium on Applications and the Internet Workshops (pp. 480–487).

Ji, S., Pan, S., Cambria, E., Marttinen, P., & Yu, P. S. (2022). A survey on knowledge graphs: representation, acquisition, and applications. IEEE Transactions on Neural Networks and Learning Systems, 33(2), 494-514. doi: 10.1109/TNNLS.2021.3070843.

Jin, Y., Ji, W., Shi, Y., Wang, X., & Yang, X. (2023). Meta-path guided graph attention network for explainable herb recommendation. Health Information Science and Systems, 11(5). doi:10.1007/s13755-022-00207-6.

Karthik, R. V., & Ganapathy, S. (2021). A fuzzy recommendation system for predicting customers' interests using sentiment analysis and ontology in e-commerce. Applied Soft Computing, 108. https://doi.org/10.1016/j.asoc.2021.107396.

Khodadadi, F., & Sinnott, R. O. (2017). A semantic-aware framework for service definition and discovery in the Internet of Things using CoAP. Procedia Computer Science, 113, 146-153.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1746–1751. ACL.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In Proceedings of the 5th International Conference on Learning Representations (ICLR) (pp. 2873-2879). Toulon, France. LeCun and Bengio, 1998

Kompan, M., & Bieliková, M. (2010). Content-based news recommendation. In Proceedings of the 11th International Conference on Electronic Commerce and Web Technologies (EC-Web), Bilbao, Spain (pp. 61-72). Springer.Langegger, A. (2005). Agent-based semantic web services. International Journal of Web Services Research, 2(2), 33-54.

Lassila, O., & Swick, R. (1999). Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation.

Li, X., Lyu, M., Wang, Z., Chen, C.-H., & Zheng, P. (2021). Exploiting knowledge graphs in industrial products and services: A survey of key aspects, challenges, and future perspectives. Computers in Industry, 129. doi: 10.1016/j.compind.2021.103449.

Lin Y, Liu Z, Luan H, Sun M, Rao S, Liu S (2015a). Modeling relation paths for representation learning of knowledge bases. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 705–714. Lisbon, Portugal: ACL.

Liu, K., Wang, F., Ding, Z., Liang, S., Yu, Z., & Zhou, Y. (2022). Recent progress of using knowledge graph for cybersecurity. Electronics, 11(15). https://doi.org/10.3390/electronics11152287.

Liu, Y., Yang, S., Xu, Y., Miao, C., Wu, M., & Zhang, J. (2021). Contextualized graph attention network for recommendation with item knowledge graph. IEEE Transactions on Knowledge and Data Engineering. Early access. doi: 10.1109/TKDE.2021.3082948.

Ma, T., Huang, L., Lu, Q., & Hu, S. (2023). KR-GCN: Knowledge-aware reasoning with graph convolution network for explainable recommendation. ACM Transactions on Information Systems, 41(1), 1-27. doi: 10.1145/3511019.

Madhu, G., Maheshwari, S., Sinha, R., Kumar, R., & Joshi, S. (2011). Ontology development for manufacturing systems using semantic web technologies. Robotics and Computer-Integrated Manufacturing, 27(3), 667-680.McKensy-Sambola, D., Rodríguez-García, M. Á., García-Sánchez, F., & Valencia-García, R. (2022). Ontology-Based Nutritional Recommender System. Applied Sciences, 12(1), 143. https://doi.org/10.3390/app12010143

Middleton, S. E., Roure, D. D., & Shadbolt, N. R. (2009). Ontology-based recommender systems. In S. Staab & R. Studer (Eds.), Handbook on Ontologies (pp. 1–8). Springer.

Mirizzi, R., Di Noia, T., Ragone, A., Ostuni, V. C., & Di Sciascio, E. (2012). Movie recommendation with DBpedia. *Proceedings of the third Italian information retrieval workshop, IIR* (pp. 101-112).

Murakami, T., Mori, K., and Orihara, R. (2007). Metrics for evaluating the serendipity of recommendation lists. In Proceedings of the Japan Society for Artificial Intelligence (pp. 137-147). Springer.

Musto, C., Semeraro, G., Lops, P., de Gemmis, M. (2013). Contextual eVSM: A content-based context-aware recommendation framework based on distributional semantics. In: Huemer, C., Lops, P. (eds) E-Commerce and Web Technologies. EC-Web 2013. Lecture Notes in Business Information Processing, vol 152. Springer, Berlin, Heidelberg, pp. 125–136. doi: https://doi.org/10.1007/978-3-642-39878-0_12,.

Nassabi, M. H., Op den Akker, H., & Vollenbroek-Hutten, M. (2014). An ontology-based recommender system to promote physical activity for pre-frail elderly. In M. Koch, A. Butz, & J. Schlichter (Eds.), Mensch und Computer 2014 Workshopband (pp. 181–184). München: Oldenbourg Wissenschaftsverlag.

Nicholson, D. N., & Greene, C. S. (2020). Constructing knowledge graphs and their biomedical applications. Computational and Structural Biotechnology Journal, 18, 1414-1428. doi: 10.1016/j.csbj.2020.05.017.

Obeid, C., Lahoud, I., El Khoury, H., & Champin, P. A. (2018). Ontology-based recommender system in higher education. Companion Proceedings of the Web Conference 2018 (pp. 1031–1034). International World Wide Web Conferences Steering Committee.

Ouf, S., Ellatif, M. A., Salama, S. E., & Helmy, Y. (2017). A proposed paradigm for smart learning environment based on semantic web. Computers in Human Behavior, 72, 796–818. https://doi.org/10.1016/j.chb.2016.08.030

Ouyang, J., Jin, Y., & Zhou, X. (2022). Dual knowledge multimodal network for recommender system. CoRR, abs/1607.0.

Page, L, S Brin, R Motwani and T Winograd (1999). The pagerank citation ranking: bringing order to the web. Technical Report (66), Stanford InfoLab, http://ilpubs.stanford.edu:8090/422/.

Palumbo, E., Rizzo, G., & Troncy, R. (2017). Entity2Rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation. In Proceedings of the 11th ACM Conference on Recommender Systems (RecSys) (pp. 32-36). Como, Italy: ACM. http://doi.acm.org/10.1145/3109859.3109889.

Park, S.-H., & Han, S.P. (2012). Empirical analysis of the impact of product diversity on long-term performance of recommender systems. In Proceedings of the 14th Annual International Conference on Electronic Commerce, June 2012, Singapore (pp. 280–281).

Patel, M., & Jain, S. (2021). Semantic web technologies for knowledge representation and reasoning: A systematic literature review. Journal of Intelligent & Fuzzy Systems, 40(4), 6729-6747.

Pazzani, M. J., & Billsus, D. (2007). Content-Based Recommendation Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds) The Adaptive Web. Lecture Notes in Computer Science, vol 4321 (pp. 325-341). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-72079-9_10

Peng, C., Xia, F., Naseriparsa, M., & Osborne, F. (2023). Knowledge graphs: Opportunities and challenges. Artificial Intelligence Review. doi: 10.1007/s10462-023-10465-9.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (pp. 701–710). New York, USA: ACM. https://doi.org/10.1145/2623330.2623732.

Piselli, A., Felici, G., Lavia, L., Fioravanti, A., Bragazzi, N. L., & Biccario, A. (2022). Semantic interoperability for personalized medicine: A systematic review. Briefings in Bioinformatics. https://doi.org/10.1093/bib/bbab076

Prud'hommeaux, E., & Seaborne, A. (2008). SPARQL query language for RDF. W3C Recommendation.

Ragab, A. H., & El-Kafrawy, P. (2022). Using knowledge graph embeddings in embedding-based recommender systems. In 2022 20th International Conference on Language Engineering (ESOLEC) (pp. 129-132). Cairo, Egypt. doi: 10.1109/ESOLEC54569.2022.10009491.

Rahayu, N. W., Ferdiana, R., & Kusumawardani, S. S. (2022). A systematic review of ontology use in e-learning recommender systems. Computers and Education: Artificial Intelligence, 3. https://doi.org/10.1016/j.caeai.2022.100047.

Ringler, D., & Paulheim, H. (2017). One knowledge graph to rule them all? Analyzing the differences between DBpedia, YAGO, Wikidata & co. In G. Kern-Isberner, J. Fürnkranz, & M. Thimm (Eds.), KI 2017: Advances in Artificial Intelligence (pp. 445-459). Springer. doi: 10.1007/978-3-319-67190-1_33.

Salter, J., & Antonopoulos, N. (2006). CinemaScreen recommender agent: Combining collaborative and content-based filtering. IEEE Intell. Syst., 21, 35–41.

Salton, G., Wong, A., & Yang, C. S. (1975). A Vector Space Model for Automatic Indexing. Communications of the ACM, 18, 613-620.

Schafer, B., Frankowski, D., Herlocker, J., Shilad, S., & Sen, S. (2007). Collaborative filtering recommender systems.

Shadbolt, N., Hall, W., & Berners-Lee, T. (2006). The semantic web revisited. IEEE Intelligent Systems, 21(3), 96-101.

Shani, G., Gunawardana, A., & Koren, Y. (2011). Evaluating Recommendation Systems. In Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (Eds.), Recommender Systems Handbook (pp. 257-297). Springer.

Shi, C., Hu, B., Zhao, W. X., & Yu, P. S. (2019). Heterogeneous information network embedding for recommendation. IEEE Transactions on Knowledge and Data Engineering, 31(2), 357-370.

Shimizu, R., Matsutani, M., & Goto, M. (2022). An explainable recommendation framework based on an improved knowledge graph attention network with massive volumes of side information. Knowledge-Based Systems, 239. https://doi.org/10.1016/j.knosys.2021.107970.

Sikos, L. F. (2023). Cybersecurity knowledge graphs. Knowledge and Information Systems, 27. doi: 10.1007/s10115-023-01860-3.

Son, J., & Kim, S. B. (2017). Content-based filtering for recommendation systems using multiattribute networks. Expert Systems with Applications, 89, 404–412.

Tarus, J. K., Niu, Z., & Kalui, D. (2018a). A hybrid recommender system for e-learning based on context awareness and sequential pattern mining. Soft Computing, 22(8), 2449–2461. https://doi.org/10.1007/s00500-017-2720-6

Tarus, J. K., Niu, Z., & Mustafa, G. (2018b). Knowledge-based recommendation: A review of ontology-based recommender systems for e-learning. Artificial Intelligence Review, 50(1), 21–48. https://doi.org/10.1007/s10462-017-9539-5.

Tiddi, I., & Schlobach, S. (2022). Knowledge graphs as tools for explainable machine learning: A survey. Artificial Intelligence, 302. doi: 10.1016/j.artint.2021.103627.

Tintarev, N., & Masthoff, J. (2015). Explaining recommendations: Design and evaluation. In Recommender Systems Handbook. Springer, 2015.

Tu, K. P., Cui, P., Wang, D., Zhang, Z., Zhou, J., Qi, Y., & Zhu, W. (2021). Conditional graph attention networks for distilling and refining knowledge graphs in recommendation. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management. Virtual Event, Queensland, Australia, November 1-5.

Vargas, S., Castells, P., & Leiva, L. A. (2011). Rank and relevance in novelty and diversity metrics for recommender systems. In Proceedings of the 5th ACM Conference on Recommender Systems (RecSys), October 23-27, 2011, Chicago, Illinois, USA (pp. 109-116). ACM.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks." Stat, 1050(20), 10-48550.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P. A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research, 11(110), 3371–3408.

Waagmeester, A., Stupp, G., Burgstaller-Muehlbacher, S., Good, B. M., Griffith, M., Griffith, O. L., Hanspers, K., Hermjakob, H., Hudson, T. S., Hybiske, K., Keenan, S., LaBonte, S., Maglott, D., Montegomery, P., & Su, A. I. (2020). Wikidata as a knowledge graph for the life sciences. eLife, 9. doi: 10.7554/eLife.52614

Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., & Guo, M. (2019a). Knowledge graph convolutional networks for recommender systems. In Proceedings of the World Wide Web Conference (WWW) (pp. 3307–3313). San Francisco, USA: ACM. doi:10.1145/3308558.3313417.

Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., & Guo, M. (2018a). RippleNet: Propagating user preferences on the knowledge graph for recommender systems. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM), pp. 417–426.

Wang, H., Zhang, F., Zhang, M. M., Leslovec, J., Zhao, M., Li, W., & Wang, Z. (2019b). Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), pp. 968–977.

Wang, J., Li, R., Tapaswi, M., Liao, R., Jia, J., Urtasun, R., & Fidler, S. (2018b). Billion-scale Commodity embedding for e-commerce recommendation in Alibaba. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), pp. 839–848.

Wang, Q., Mao, Z., Wang, B., & Guo, L. (2017). Knowledge graph embedding: A survey of approaches and applications. IEEE Transactions on Knowledge and Data Engineering, 29(12), 2724–2743.

Wang, X., He, X., Cao, Y., Liu, M., & Chua, T. S. (2019b). KGAT: Knowledge graph attention network for recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), pp. 950–958.

Wang, X., Huang, T., Wang, D., Yuan, Y., Liu, Z., He, X., & Chua, T.-S. (2021). Learning intents behind interactions with knowledge graph for recommendation. In Proceedings of the Web Conference, April 19-23, (pp. 878-887). Ljubljana, Slovenia.

Wang, X., X. He, Y. Cao, M. Liu, and T. S. Chua (2019c). KGAT: knowledge graph attention network for recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), pp. 950–958. Anchorage, USA, ACM.

Wang, Z., Zhang, J., Feng, J., & Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI), pp. 1112–1119.

Weihong, H., & Yi, C. (2006). An e-commerce recommender system based on content-based filtering. Wuhan Univ. J. Nat. Sci., 11, 1091–1096.

Wen, X. (2021). Using deep learning approach and IoT architecture to build the intelligent music recommendation system. Soft Computing, 25, 3087–3096.

Wu, T., Khan, A., Yong, M., Qi, G., & Wang, M. (2022). Efficiently embedding dynamic knowledge graphs. Knowledge-based systems, 250. https://doi.org/10.1016/j.knosys.2022.109124.

Xiaoyan, C., Yu, Z., Libin, Y., Tao, D., & Lantian, G. (2018). Bibliographic network representation based personalized citation recommendation. IEEE Access, 1-1.

Xu, F., Lian, J., Han, Z., Li, Y., Xu, Y., & Xie, X. (2019). Relation-aware graph convolutional networks for agent-initiated social e-commerce recommendation. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM) (pp. 91-100). Beijing, China, November 3-7, ACM. doi:10.1145/3357384.3357924.

Yang, Z., & Dong, S. (2020). HAGERec: Hierarchical attention graph convolutional network incorporating knowledge graph for explainable recommendation. Knowledge-Based Systems, 204. https://doi.org/10.1016/j.knosys.2020.106194

Yang, Z., & Dong, S. (2020). HAGERec: Hierarchical attention graph convolutional network incorporating knowledge graph for explainable recommendation. Knowledge-Based Systems, 204.

Yin, L., Zhong, R. R., & Wang, J. (2023). Ontology based package design in fresh e-commerce logistics. Expert Systems with Applications, 212.

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD'18), pp. 974–983. London, UK: ACM.

Zehra, S., Wasi, S., Jami, I., Nazir, A., Khan, A., & Waheed, N. (2017). Ontology-based sentiment analysis model for recommendation systems. In Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2017), Volume 2, pp. 155–160.

Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W.-Y. (2016). Collaborative knowledge base embedding for recommender systems. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 353-362. San Francisco, USA: ACM.

Zhang, M., Hurley, N., & O'Mahony, M. P. (2008). Avoiding monotony: Improving the diversity of recommendation lists. In Proceedings of the 2nd ACM Conference on Recommender Systems (RecSys), October 23-25, 2008, Lausanne, Switzerland (pp. 123-130). ACM.

Zhang, Y., Ai, Q., Chen, X., & Wang, P. (2018). Learning over knowledge-base embeddings for recommendation. arXiv preprint arXiv:1803.06540.

Zhu, H. (2022). MetaAID: A flexible framework for developing metaverse applications via AI technology and human editing. arXiv preprint arXiv:2204.01614. https://doi.org/10.48550/arxiv.2204.01614.

Zhu, X., Guo, S., Zhang, W., & Liu, H. (2017). Multi-modal knowledge graph construction and application: A survey. IEEE Transactions on Knowledge and Data Engineering. doi: 10.1109/TKDE.2022.3224228.

Zou, X. (2020). A survey on application of knowledge graph. Journal of Physics Conference Series, 1487(1), 012016. doi: 10.1088/1742-6596/1487/1/012016.