Explanations in Recommender Systems using Linked Open Data - A Survey

Mohammed Alshammari†
mohammed.alshammari@nbu.edu.sa
†Department of Computer Sciences, Faculty of Computing and Information Technology
Northern Border University, Rafha, Kingdom of Saudi Arabia

Summary
Explanations in recommender systems became a hot research topic due to the abundance of data on today's internet.
These explanations are proven to increase the transparency of the system and gain user trust. However, the data sparsity issue makes generating explanations a challenge and requires more meaningful data. Therefore, other information sources, such as linked open data (LOD), can play a key role in solving the problem. In this paper, we will examine the literature to find the impact of LOD on enhancing the process of generating explanations for recommender systems.

Keywords:
Artificial Intelligence, Machine Learning, Recommender Systems, Explanations, Linked Open Data, Semantic Web.

1. Introduction

Recommender systems have become crucial in today's research field due to the abundance of data on the internet. In addition, the business field requires advances in recommender systems methodologies to meet their needs with processing huge quantities of data and even increasing their revenues. Recommender systems have proved to have the ability to handle unrelated data and still output meaningful information. Business giants (e.g., Amazon) have relied heavily on recommendation engines to provide better services and hence increase their market value, especially during the Covid- 19 period when people relied more on online shopping and flooded the internet with even more data. It is well known that more data being fed to recommender systems leads to improved performance and output; however, increasing users' trust in the output of recommender systems is a challenging task that requires more effort. Consequently, the need for explanations arises. Explanations have proved to be effective at increasing the transparency of recommendation engines. They can come in many forms, such as plain text, tags, graphs, and pictures. Amazon uses a plain text form for explaining recommendations, such as "People who bought this product also bought this other product." In content-based filtering, the explanation generation is a quite straightforward process. However, in collaborative filtering, especially when data about either items or users are few and sparse, the task becomes challenging. Therefore, another source of information is needed to help enhance the process of generating meaningful, yet convincing, interpretations.

Linked open data (LOD) is a platform for linking data together meaningfully so that machines can auto-process them. LOD provides rich information that can be used to compensate for the shortage of information in such domains (e.g., rating movies) so the accuracy of recommendations can be improved in addition to facilitating an explanation generation procedure that will result in increased transparency of the system.

In this study, we will review the literature on the use of LOD in the explanation generation process toward generating more trustworthy recommendations. By surveying the literature, we should be able to answer the following two questions:
(1) Did including rich information from LOD lead to better recommender systems in general? (2) Did it lead to increased transparency by generating more convincing explanations? We will try to answer these questions in this paper.

We limited the search scope to include only research conducted within the last six years, since 2016. The reason for this was to focus on the latest updates on this topic. We used Saudi Digital Library (SDL)[1] and Google Scholar[2] as the main sources for the search. The list of words used in the search included: recommender system, recommendation engine, linked open data, semantic web, ontology, explanations, interpretations, and justifications.

2. Literature on the Use of Linked Open Data (LOD) in Recommender System Explanations

Musto et al. [1] used available properties in LOD (e.g., author, director, etc.) to justify the outcome of a recommender system model. Properties offer a rich description of items in the movie domain, as an example. Therefore, they help in generating explanations for such recommendations. The authors [1] claimed that most of the

Manuscript received May 5, 2022
Manuscript revised May 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.5.76

earlier efforts in the recommendation engines development area focused on improving the accuracy of the recommendation, while neglecting the aspect of explanation. To overcome this gap, natural language interpretation generated from LOD provided a decent solution to balance between accuracy and transparency. In this approach, the first step was mapping all unique items in the dataset to those in the LOD cloud, aiming to extract the desired properties for the explanation generation process. The next step was constructing a graph that linked all liked items with a list of recommended items that the recommendation engine produced for a particular user. The graph captures both direct and indirect relationships between items. Ultimately, the explanation generation process resulted in a higher level of transparency [1].

The explanation process goes through four main steps. The first is mapping, which focuses on linking items, either liked by the user or recommended to them, with the corresponding URL in LOD. The goal is to take advantage of available properties in the process of building explanations. After mapping is complete, the builder process constructs a representative graph that holds all descriptive properties. The explanation generation process relies strongly on this step because it helps to map users' liked items to recommended items through shared features obtained from the first step [1].

In greater detail, the first step is the Mapper, which is where the linking of current items in the user profile with the corresponding item in LOD occurs. In this work, DBpedia ontology is used in addition to SPARQL[3], the query language for ontology, to retrieve the desired information from DBpedia. For each item in the user profile, a set of representations from DBpedia is collected and fed to the Builder. Next, a graph is constructed and set out as the main determination for the explanation generation process. Two primary kinds of builders are proposed: the first concentrates on the direct properties linking items, such as the lead actor in a movie, and the second extends the Builder to include indirect properties. For example, if a user likes movies that share certain properties such as being filmed in the 1990s films or American films, then that information can be used to build broader justifications for recommendations [1].

After Builder provides explanations for the recommendations, which consequently feed the Ranker, a relevance score is given to the explanations, helping to provide the most relevant explanations for the recommendation. The formula is as follows: [1]:

$$score(p, I_u, I_r) = \left(\alpha \frac{n_{p,I_u}}{|I_u|} + \beta \frac{n_{p,I_r}}{|I_r|}\right) * IDF(p) \quad (1)$$

Here, p is a given property, I_u is a set of items the user liked before, and I_r denotes a set of recommended items for a certain user. $n_{p,Iu}$ represents the connection of p and I_u, and $n_{p,Ir}$ is the number of connections between p and I_r. α and β are controlling factors. *IDF* is the famous inverse document frequency technique [2], which totals the number of items the target property has been associated with throughout the complete dataset.

Another formula is designed to calculate the total score of all indirect properties, denoted as b, that link all items in the user profile and recommendations, as follows [1]:

$$score(b, I_u, I_r) = \sum_{i=1}^{|P_c(b)|} score(p_i, I_u, I_r) * IDF(b) \quad (2)$$

All properties, direct and indirect, are associated with a relevance score and then ranked to top_K for justification generation [1]. Finally, the generator step generates a natural language interpretation for the recommendations, following either the direct property technique explained earlier and denoted as $p \in P$, or the indirect one denoted as $p \in P_b$ [1].

For the evaluation process, three different domains are used to examine the proposed work: music (Last.fm[4]), movies (MovieTweetings[5]), and books (DBook[6]).

For the recommendation technique used in this work, three kinds of algorithms were implemented: Personalized PageRank (PPR) [3], content-based recommender systems (CBRS), and collaborative filtering (CF).

A user study involving 680 individuals was conducted to answer the main research questions, which included measuring the effectiveness of the explanations after exploiting LOD and examining the independence of the algorithm in the explanation generation process. Finally, the proposed work was tested in terms of producing satisfying justifications for n number of recommendations.

The sample group evaluated the proposed explanation generation technique and completed a questionnaire to measure the transparency, persuasion, engagement, trust, and effectiveness of the work. The experimental results for all three domains, movies, books, and music, indicated that the use of LOD in explanations increased the transparency and trust in the recommender system that relied on PPR and CF. Moreover, engagement and effectiveness measures rose positively when PPR was used. For recommender systems using the CBRS algorithm, the results were a slight increase in transparency measure only.

Musto et. al. [4] proposed a novel method that exploits the user's reviews of items in generating explanations. In this work, the authors proposed a framework that generates

independent natural language explanations by introducing the recommended item alongside the reviewed items to the model.

Three different applications were proposed for the purposes of validation. First, a combination of natural language processing (NLP) and sentiment analysis methods were used to generate the most relevant reviews for the justification process, which was denoted as NLP-PIPELINE. The second method added to the first one some different aspects assignments to the recommended items, denoted as TS-PIPELINE. The last technique handled the issue of explanation context-awareness by making a lexicon that allows for learning about the context of such a recommendation, represented as CONTEXT- PIPELINE [4]. The authors claimed that the novelty of the proposed work came from the explanation process being independent of the recommendation process. It used the reviews of items in addition to the recommendation list in feeding the model, which then generated natural language interpretations.

The first method, NLP-PIPELINE, consists of three stages. The first stage is aspect extraction, where aspects of the recommended item are extracted using the user's reviews. In the next step, the extracted aspects are ranked on the basis of relevancy to the recommended items using the following formula:

$$score(a, R_i) = \left(\alpha \frac{rel_{a,R_i}}{|S_{R_i}|} + \beta \frac{pos(\alpha,R_i)}{pos(\alpha,R_i)+neg(\alpha,R_i)}\right) * IAF(a, R_i) \quad (3)$$

where a represents a particular aspect, R_i is the set of available reviews for item i, and SR_i denotes the sentences generated from reviews. rel_a, represents the total number of sentences relevant to aspect a in the set R_i, $pos(\alpha, R_i)$ and $neg(\alpha, R_i)$ denote the positivity and negativity of extracted sentences in regard to the recommended items, obtained using a sentiment analysis technique. α and β are smoothness controlling variables. *IAF* stands for inverse aspect frequency, which is inspired by *IDF*, the famous method that puts more weight on the most relevant items despite frequency. The last stage is generating the justification using a template-based design and filling it with the most relevant aspects extracted from the previous stage [4].

There are two drawbacks to this method. First, it simply counts aspects, taking advantage of *IDF*. Second, it uses a static template for the justification generation process. Thus, a solution is proposed in the TS-PIPELINE method for extracting more aspects related to the recommended items in the latter two stages of the previous method. For aspect ranking, the following formula is used:

$$score(a, R_i) = \left(\alpha \frac{n_{a,R_i}}{|S_{R_i}|} + \beta \frac{pos(\alpha,R_i)}{pos(\alpha,R_i)+neg(\alpha,R_i)}\right) * rel_{a,R_i} \quad (4)$$

Here, $rel_{a,Ri}$ is replaced by $n_{a,Ri}$ which denotes the simple count of aspects, and *IDF* is replaced by the Kullback–Leibler divergence to give a higher score for more closely related aspects.

In the third stage, the goal is to generate a more dynamic justification rather than a static one, as in the previous method. Therefore, the text summarization technique [5] is used.

The last method developed in this work is CONTEXT-PIPELINE, which solves the issue with the first method, the lack of diversity of explanations. In this method, the idea is inspired by [6] and [7], where the user's decision about such a recommendation list is influenced by certain factors or aspects. In the aspect extraction phase, a lexicon of items' aspects is learned and then used to generate the justification. The formula for aspect ranking is as follows:

$$score(a, R_i) = \left(\alpha \frac{rel_{a,R_i}}{|S_{R_i}|} + \beta \frac{pos(\alpha,R_i)}{pos(\alpha,R_i)+neg(\alpha,R_i)} + \gamma \frac{n_{a,c}}{|S_c|}\right) * IAF(a, R_i) \quad (5)$$

where c denotes the contextual setting, and $\frac{n_{a,c}}{}$ represents the percentage of all sentences to the contextual setting.

In the evaluation step, the authors evaluated five metrics–transparency, engagement, effectiveness, trust, and persuasiveness–by conducting a user study. A sample of 286 people were involved in this study. For the dataset, MovieLens and BookCrossing were used. A mapping from both datasets to DBpedia was carried out, limiting the number of items to those that existed in both sources. Amazon reviews about movies and books in the dataset were included for the aspect extraction process. For model performance comparison, ExpLOD [8] was used.

Multiple experiments were conducted, comparing the performance of NLP-PIPELINE to ExpLOD, NLP-PIPELINE to TS-PIPELINE, and NLP-PIPELINE to CONTEXT-PIPELINE. Results showed that NLP-PIPELINE outperformed ExpLOD in all metrics in both the movies and books domains. However, TS-PIPELINE performed better than NLP-PIPELINE using all five metrics. Finally, CONTEXT-PIPELINE's performance resulted in much higher levels of satisfaction by users for all metrics than did NLP-PIPELINE's performance

A theoretical paper that exploited semantic web technologies for increasing users' trust in the system was proposed by [9] [10]. The goal of this work is to build an ontology that provides descriptions for recommendation explanations. Four phases were proposed to formalize the explanations ontologically: motivation, knowledge container, generation, and presentation [9]. Out of many ontological concepts, three were used in this formalization technique, which were composition, sub-concept, and instance.
Explanations may be categorized following the motive and goal of the user. Therefore, the explanation is supposed to meet user expectations regarding increasing their satisfaction, system transparency, efficiency, scrutability, trust, and persuasiveness. The second phase is the knowledge containers, where recommendation interpretations are fed from different sources of knowledge, which can be either from the dataset or the output of the recommendation engine itself. Next is the generation process, where explanation generation occurs following two methodologies: introspective when the recommender system is a white box, and external when it applies the black box technique. Finally, the presentation of the justification is carried out using any of multiple formats, such as natural language explanations, visuals, schematics, and others [9]. Also, the argumentation direction (e.g., positive, negative, similarities, differences) is considered when generating the explanations.

Lully et. al. [11] claimed that in the literature, there exist many recommender system models in the travel domain; however, they lack explanations, especially using LOD. Therefore, they proposed a model that facilitates the recommendation engine with explanations that are enhanced by the power of LOD. This study focused on exploiting the item's description in the process of generating explanations. In particular, it tried to solve three issues: filtering entities, missed user-friendliness, and less intelligent justification. For each of the above-mentioned shortcomings, the authors proposed such solutions, using the DBpedia encyclopedia for the entity filtering issue and to increase the intelligibility of the system. Also, they tackled the third shortcoming by leveraging the textual description in composing better sentences for the explanations.

The entity filtering stage was completed by giving a relevant degree for the subject of the entity to the travel domain. A tree was constructed using information from DBpedia, which contains over 1 million categories spread across 15 levels. Starting from the second level, which contains 43 categories, manual annotation was carried out to filter for only the categories related to the travel domain, minimizing the number of categories to 12. In a cascading manner, all subcategories of those 12 were included in the process. As a result, any entity obtained from DBpedia where the description contained any of the filtered categories would be considered relevant. Regarding the intelligibility of the system, the authors suggested that including more knowledge resources, such as YAGO and Schema.org, would help to find more descriptive details for such entities. This inclusion allowed for the creation of a more user-friendly justification by showing meaningful sentences instead of a list of entities.
The sentence generation process included applying some well- known techniques, such as TF-IDF and tokenization, for improved output.

For evaluation, they conducted a user study asking 30 participants their thoughts about the system to measure five metrics: efficiency, satisfaction, effectiveness, intelligibility, and relevance. Four explanation generation methods were compared in this experiment: entity-based (EN), sentences in natural language (NL), pure class based (PC), and companion class-based (CC). The results showed a clear advantage for the proposed NL method over the other methods in all five metrics [11]. This proves that exploiting the power of LOD, especially the description side of the items, increases the overall performance of the model.

Alshammari et al. [12] proposed a black box model using a matrix factorization technique [13] that leverages LOD to generate explanations. They claimed that producing explanations for the black box recommender system was challenging because building predictions occurs in the hidden latent spaces [12]. Collaborative filtering algorithms are used to find similarities between users based on available information, such as ratings. Techniques like the matrix factorization use ratings only to predict the rating of unseen items by taking advantage of hidden features in the latent spaces. As a consequence of this shortage of information, generating explanations becomes more challenging [12]. The proposed solution was to introduce LOD as an additional regularization term to the objective function of the matrix factorization technique, which made generating explanations a straightforward process. As a result, accuracy and transparency increased.

The side information of items was acquired from the DBpedia encyclopedia through SPARQL queries and then preprocessed to be introduced to the model [12].
Following is the objective function:

$$J = \sum_{u,i \in R} \left(R_{u,i} - p_u q_i^T\right)^2 + \frac{\gamma}{2} \sum_{i,j \in S^{ldsd}} \left(S_{i,j}^{ldsd} - q_i q_j^T\right)^2 + \frac{\beta}{2} (\| p_u \|^2 + \| q_i \|^2). \quad (6)$$

where u is the user, i is the item, and $R_{u,i}$ is the actual rating. p_u denotes the latent factor of users, and q_i is the latent factor for items. S^{ldsd} represents a knowledge graph derived from LOD using SPARQL language and used to feed the model with side information about items, which then facilitates the

model with explanations. q_i and q_j represent items from the LOD. γ and β are two smoothing coefficients.
For updating rules, a stochastic gradient descent is applied for the goal of the convergence of p and q, as follows:

$$p_u^{(t+1)} \leftarrow p_u^{(t)} + \alpha\left(2\left(R_{u,i} - p_u^{(t)}\left(q_i^{(t)}\right)^T\right)q_i^{(t)} - \beta p_u^{(t)}\right) \quad (7)$$

$$q_i^{(t+1)} \leftarrow q_i^{(t)}\alpha + 2\left(R_{u,i} - p_u^{(t)}\left(q_i^{(t)}\right)^T\right)p_u^{(t)} + 2\gamma\left(S_{i,j}^{ldsd} - q_i^{(t)}\left(q_j^{(t)}\right)^T\right)q_j^{(t)} - \beta q_i^{(t)}. \quad (8)$$

The main contribution of this work is the addition of an item-to-item similarity and using that to enhance the building of the model in the low-dimensional latent spaces, which results in the addition of explainability to the output of the black box system. The authors examined their methodology using the MovieLens dataset in addition to DBpedia for the item's side information. The experiments showed that their model outperformed the baselines using different metrics, such as root mean square error, precision, and recall. In addition, they conducted a user study involving 34 participants. Three metrics are measured in this study, transparency, satisfaction, and effectiveness. All three metrics recorded high numbers after the analysis of the participants' answers to the questionnaire following their experience of the system.

Fernando et al. [14] presented a music recommendation model that interacts with the user through a chatbot in Telegram and builds an ontology that links users' interests with data from the Kaggle website. After feeding the ontology with both the user interests in music and Kaggle, the system recommends a music item with a template-ready explanation using the relationships between ontology's classes. In evaluation, a user study was conducted with 63 participants aged between 16 and 25, and the music list ranged between the years 2010 and 2019. Seven metrics were measured in this study: persuasiveness, trust, satisfaction, efficiency, effectiveness, transparency, and scrutability. The results were that a high percentage of users enjoyed the recommendations with explanations more than mere recommendations. This leads to the conclusion that explanations, especially when LOD technologies are included, play a major role in refining the recommendation process [14].

Another paper that succeeded in leveraging LOD in the process of building recommendations was by [15], who proposed a touring model that suggests points of interest (POI) by taking advantage of social media profiles of the target user in addition to the information available in LOD, particularly, Europeana[7] and DBpedia[8]. The system consisted of four main steps. First, the system retrieved information from Facebook for the target user. Second, it used LOD to enhance the model with more information, which led to the construction of a social graph. Finally, it generated a list of cultural recommendations in the area where the target user was located [15]. For the recommendation engine, the authors developed the Cicero algorithm [16] [17], a social recommendation engine, by introducing LOD to it. This engine consists of three phases: (1) social information retrieval, (2) constructing the user model, and (3) recommending items.

Facebook, the giant social network that provides social activities related to locations, was used in this work through its graph API to gather information about users. The user model was then built by constructing a graph where each node was represented by four main classes: person, place, location, and category [15]. Relationships between the nodes were confined to one of the following properties: has-category, located, visited, and knows. The cicero algorithm was applied to build a recommendation list in two ways, socially and semantically. For the social recommender system, categories from Facebook that are not related to the touring domain were filtered out, reducing the number from 238 to 37. Consequently, only interactions of users that included those 37 categories were kept in the process of building the model. The goal was to recommend to the user new places that their friends had visited, following the notion that a friend's interest is trustworthy [15]. For the semantic recommender system, two LOD platforms, Europeana and DBpedia, were used to enrich the system with related data. A SPARQL query was formed to collect valuable information from DBpedia, in addition to URIs, for further information. Also, a list of possibly likable places was created from Europeana. Combining the previous two methods resulted in suggesting a recommendation that has a high degree of semantic similarity. Passant [18] proposed a method for calculating the semantic similarity, which was adapted in this work. There were three kinds of similarities: direct, indirect, and a combination of the two.

For the evaluation process, several baseline methods were used. The *nDCG* metric at three degrees–1, 5, and 10–was used to evaluate the model performances. Results show that the proposed work outperformed the baselines. In addition, a user study involving 50 participants from Facebook was conducted to test the model output. Serendipity, diversity, and novelty were measured, and the results showed that the proposed method obtained the highest degree of serendipity. However, its diversity was among the lowest, while the degree of novelty was moderate in comparison to other baselines.

[7] https://www.europeana.eu/
[8] https://wiki.dbpedia.org/

3. Conclusion

According to the previous works that we reviewed in this survey, it appears that taking advantage of the rich information that LOD provides surely improves the accuracy and transparency of recommender systems. The use of LOD differs based on the type of recommender system algorithm (e.g., content-based filtering, collaborative filtering, etc.) and the domain of the model (e.g., travel, books, movies, music, etc.) where the generation of persuasive explanations is intended. Taking advantage of LOD, explanations come in different forms. They may come in a natural language format, as in [1], [4], [11], and [14]; an ontology format, as in [9] and [10]; or a graph format, as in [12] and [15]. Overall, the studies in the literature prove that the inclusion of LOD in the process of building a recommender system model helps increase the accuracy and transparency measures, which in turn increases the satisfaction and trust of the user in the system.

References

[1] Cataldo Musto, Fedelucio Narducci, Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. Linked open data-based explanations for transparent recommender systems. International Journal of Human-Computer Studies, 121:93–107, jan 2019.

[2] CD Manning, P Raghavan, and HS Schutze. Term weighting, and the vector space model. Introduction to information retrieval, pages 109–133, 2008.

[3] Taher H Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. IEEE transactions on knowledge and data engineering, 15(4):784–796, 2003.

[4] Cataldo Musto, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. Generating post hoc review-based natural language justifications for recommender systems. User Modeling and User- Adapted Interaction, 31(3):629–673, jun 2020.

[5] Cataldo Musto, Gaetano Rossiello, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. Combining text summarization and aspect-based sentiment analysis of users' reviews to justify recommendations. In Proceedings of the 13th ACM Conference on Recommender Systems, RecSys '19, page 383–387, New York, NY, USA, 2019. Association for Computing Machinery.

[6] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In 1994 First Workshop on Mobile Computing Systems and Applications. IEEE, dec 1994.

[7] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In Recommender Systems Handbook, pages 191–226. Springer US, 2015.

[8] Cataldo Musto, Fedelucio Narducci, Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. ExpLOD. In Proceedings of the 10th ACM Conference on Recommender Systems. ACM, sep 2016.

[9] Marta Caro-Martínez, Guillermo Jiménez-Díaz, and Juan A. Recio-García. Conceptual modeling of explainable recommender systems: An ontological formalization to guide their design and development. Journal of Artificial Intelligence Research, 71:557– 589, jul 2021.

[10] Marta Caro-Martinez, Guillermo Jimenez-Diaz, and Juan A Recio-Garcia. A theoretical model of explanations in recommender systems. In ICCBR, volume 52, page 2018, 2018.

[11] Vincent Lully, Philippe Laublet, Milan Stankovic, and Filip Radulovic. Enhancing explanations in recommender systems with knowledge graphs. Procedia Computer Science, 137:211–222, 2018.

[12] Mohammed Alshammari, Olfa Nasraoui, and Scott Sanders. Mining semantic knowledge graphs to add explainability to black box recommender systems. IEEE Access, 7:110563–110579, 2019.

[13] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. Computer, 42(8):30–37, aug 2009.

[14] Gerry Fernando, Z. K. A. Baizal, and Ramanti Dharayani. Music recommendation using conversational recommender system with explanation facility. In 2021 International Conference on Data Science and Its Applications (ICoDSA). IEEE, oct 2021.

[15] Giuseppe Sansonetti, Fabio Gasparetti, Alessandro Micarelli, Federica Cena, and Cristina Gena. Enhancing cultural recommendations through social and linked open data. User Modeling and User-Adapted Interaction, 29(1):121–159, mar 2019.

[16] Claudio Biancalana, Fabio Gasparetti, Alessandro Micarelli, and Giuseppe Sansonetti. An approach to social recommendation for context-aware mobile services. ACM Transactions on Intelligent Systems and Technology, 4(1):1–31, jan 2013.

[17] Alessio De Angelis, Fabio Gasparetti, Alessandro Micarelli, and Giuseppe Sansonetti. A social cultural recommender based on linked open data. In Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization. ACM, jul 2017.

[18] Alexandre Passant. Measuring semantic distance on linking data and using it for resources recommendations, 2010.

Mohammed Alshammari received the Bachelor of Education degree in computer from the University of Hail, Saudi Arabia, in 2008, and the M.Sc. degree in computer science from the University of Leicester, U.K., in 2011. And the Ph.D. degree in Computer Science from the Knowledge Discovery and Web Mining Laboratory, Computer Engineering and Computer Science Department, University of Louisville, KY, USA. In 2019. Since 2019 he works as an assistance professor at the Northern Border University, Saudi Arabia.