HSCRD: Hybridized Semantic Approach for Knowledge Centric Requirement Discovery

Rituraj Ojha[1] and Gerard Deepak[2](✉)

[1] Department of Metallurgical and Materials Engineering, National Institute of Technology, Tiruchirappalli, Tiruchirappalli, India

[2] Department of Computer Science and Engineering, National Institute of Technology, Tiruchirappalli, Tiruchirappalli, India

gerard.deepak.christuni@gmail.com

Abstract. There is a necessity for a requirement recommendation system that can eliminate the old tedious recommendation discovery process. In this paper, a hybrid semantic approach for knowledge-centric requirement discovery has been proposed. The proposed HSCRD framework takes stakeholders' interactions and preprocesses it. The individual keywords obtained are input into the TF-IDF model to yield the documents from the Requirement Specification Document Repository. The index words of these documents are extracted and are linked with Upper Domain Ontologies. The ontologies are grown by computing the semantic similarity measures, namely, Jaccard similarity and SemantoSim similarity. These grown ontologies are submitted to the Wikidata API, Freebase API, and DBPedia API to yield the Enriched Domain Ontologies. The Enriched Domain Ontologies, as features, are passed into Bi-gram and Tri-gram models. Using Bi-gram and Tri-gram of these features, input is given to the Bagging model for classification. Bagging is chosen with SVM and a highly complex Decision Tree classifier. Finally, the recommended documents and ontologies features are passed individually with respect to the classified documents through TF-IDF and semantic similarity pipeline in order to recommend the individual requirements. The proposed HSCRD achieves the highest average Accuracy with the Precision of 93.18%, Recall of 95.69%, Accuracy of 94.43%, F-Measure of 94.42%, a low FDR of 0.07, and a very high nDCG of 0.96.

Keywords: Ontologies · Requirement discovery · Requirement engineering · Semantic similarity

1 Introduction

Requirement engineering is a process in software engineering that consists of several activities, including defining the requirements, documenting them, and maintaining them. Requirement elicitation is one of the main activities of requirement engineering. It includes gathering information about the project from customers, interviews, manuals, similar software, and other project stakeholders. The process may appear simple,

S. Motahhir and B. Bossoufi (Eds.): ICDTA 2022, LNNS 455, pp. 70–79, 2022.
https://doi.org/10.1007/978-3-031-02447-4_8

but it has several underlying challenges. Firstly, the customer may not properly understand their needs as per the technologies available. They might get confused when the software is complex. Secondly, they might face difficulty communicating the specific requirements with the software engineer present. Lastly, the customer may describe unnecessary details or omit important information during requirement elicitation.

There is a need for a requirement recommendation system to automate the requirement engineering process by eliminating the long old process of tedious meetings. Also, the ontologies and real-world knowledge bases can be mapped to yield the requirements with more accuracy. To align the recommended requirements to stakeholders' needs, the recommender system should analyze previous stakeholders' interactions and old validated requirement engineering reports.

Motivation: Requirement analysis is quite tedious, and recommending micro-requirements for a specific set of stakeholders' requirements is definitely a tedious task. Therefore, mapping the requirements of similar stature via ontologies from the already existing and validated requirement engineering and requirements analysis reports is the best-chosen practice. Instead of returning the entire requirements, semantic infused learning based on ontology-based criteria mapping and learning based on the users' opinions on these ontologies is one of the best-suited methods.

Contribution: HSCRD, which is a hybridized semantic approach for knowledge-centric requirement discovery, is proposed. The stakeholders' interactions are recorded and preprocessed, and the individual keywords are input using the TF-IDF model to yield the documents from the Requirement Specification Document Repository. The index words of these documents are extracted and are linked with Upper Domain Ontologies. The ontologies are grown by computing the semantic similarity measures, namely, Jaccard similarity and SemantoSim similarity. These grown ontologies are submitted to the Wikidata API, Freebase API, and DBPedia API to yield the enriched domain ontologies. The enriched domain ontologies, as features, are passed into Bi-gram and Tri-gram models. Using Bi-gram and Tri-gram of these features, input is given to the Bagging model for classification. Bagging is chosen with SVM and a highly complex Decision Tree classifier. Finally, the recommended documents and ontologies features are passed individually with respect to the classified documents through TF-IDF and semantic similarity pipeline in order to recommend the individual requirements. The values of metrics like, Precision, Accuracy, Recall, F-Measure, and nDCG are increased.

Organization: The flow of the remaining paper is as follows. A condensed summary of the related works is provided in Sect. 2. Section 3 depicts the architecture of the proposed system. Section 4 describes the implementation and the evaluation of performance. The conclusion of the paper is presented in Sect. 5.

2 Related Works

Some of the work related to the proposed approach is discussed in this section. AlZu'bi et al. [1] have proposed a requirement recommender system which is based on Apriori algorithm. The algorithm helps in extracting rules from the user requirements which can be used to recommend requirements to stakeholders. Elkamel et al. [2] have proposed a UML based recommender system which suggests items based on the output UML class. The UML classes are classified from UML classes diagrams and contents are recommended to users based on the classified classes.

Williams [3] has proposed a recommendation system for security requirement elicitation. The paper presents the ontology-based recommender system and also conducts a study on stakeholders based on the proposed system. Avdeenko and Pustovalova [4] have proposed a methodology for assisting the requirement specification through the help of ontologies. Classes of the ontology are requirement types and instances are requirement statements.

Rajagopal et al. [5] have proposed an approach for requirement elicitation during software development. Their model helps in training the stakeholders about the capacity of the software and hardware. The model also gathers the stakeholders' conversations to extract keywords and the keywords are used for generating the requirements. They have also used methodologies namely, Quality Function Deployment and Capability Maturity Model to evaluate their results. Shambour et al. [6] have proposed a recommender system for requirement elicitation based on collaborative filtering. Their proposed model reduces the amount of time taken to go through the big requirement repositories by extracting the reusable and important requirements. In [7–20] several approaches in support of the literature of the proposed model have been depicted.

3 Proposed System Architecture

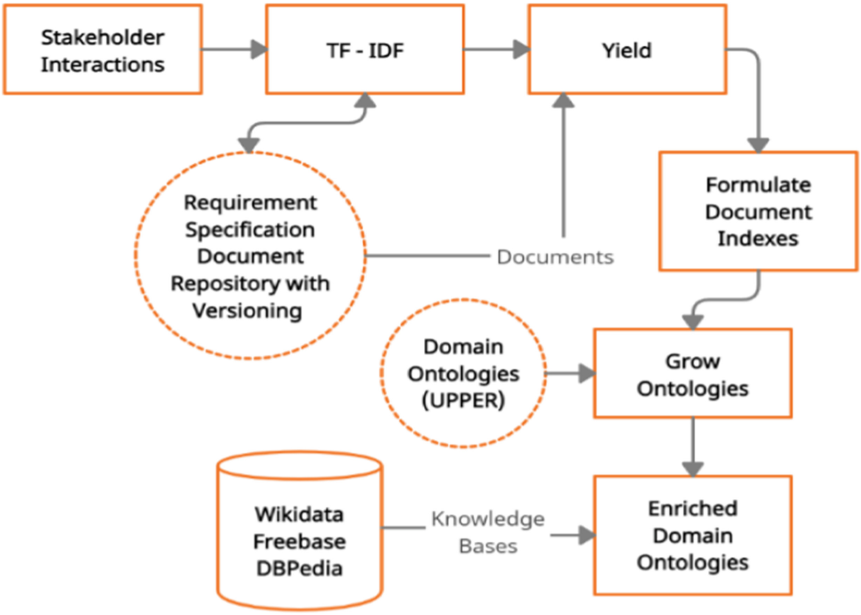
Stakeholder Interactions
TF - IDF
Yield
Requirement Specification Document Repository with Versioning
Documents
Formulate Document Indexes
Domain Ontologies (UPPER)
Grow Ontologies
Wikidata Freebase DBPedia
Knowledge Bases
Enriched Domain Ontologies

Fig. 1. Phase 1 of the proposed system architecture

The architecture for the proposed system is depicted in Fig. 1 and Fig. 2. The stakeholders' interactions are recorded and preprocessed using Tokenization, Lemmatization, stop word removal, and Named Entity Recognition (NER). Tokenization is a process of splitting the texts in the dataset into pieces called tokens. Byte Pair Encoding (BPE) is used for the tokenization process. Lemmatization involves grouping together several inflected kinds of the same word so that they can be analyzed as a single term. During stop word removal, the common ubiquitous words are removed as they add no value for the analysis and only increase the dimension of the feature set. NER is the process of finding and categorizing the data or entity into predefined categories. After preprocessing, the individual keywords are input into the TF-IDF (Term Frequency-Inverse Document Frequency) model. The TF-IDF helps in finding the importance of a term in the document. It is used to yield the documents from the Requirement Specification Document Repository which uses versioning and control mechanisms. One of the famous Software Requirements Specification (SRS) repositories is SVN. Every organization has their own or use a requirement specification document repository for versioning and control, which is known as change management or configuration management. For configuration management and change management, the requirement document will be versioned and from that repository, the final version of each document is yielded based on the TF-IDF score. TF-IDF uses the concept of the frequency of occurrence and rarity of occurrence of the word over a document corpus.

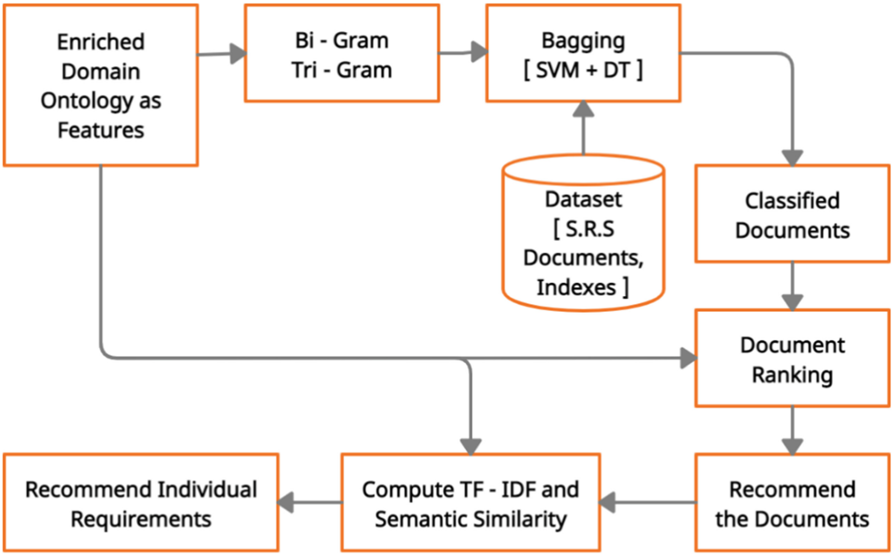
Enriched Domain Ontology as Features
Bi - Gram
Tri - Gram
Bagging
[SVM + DT]
Dataset
[S.R.S Documents, Indexes]
Classified Documents
Document Ranking
Recommend the Documents
Compute TF - IDF and Semantic Similarity
Recommend Individual Requirements

Fig. 2. Phase 2 of the proposed system architecture

From the yielded documents, the index words of these documents are extracted and are linked with Upper Domain Ontologies or the Core Domain Ontologies. The indexes are yielded from the documents, by taking keywords from each page of the document and by giving frequent and rare terms more importance. Static Upper Ontologies which are relevant to the domain of the stakeholders' interaction are passed into the framework. So, the ontologies are grown by computing the semantic similarity. The semantic similarity measure used is Jaccard Similarity and SemantoSim Similarity. Parallelly, these grown ontologies are submitted as input to the Wikidata API, Freebase API, and DBPedia API to yield the enriched domain ontologies.

The Enriched Domain Ontologies are passed into Bi-gram and Tri-gram models. Using Bi-gram and Tri-gram of these features, input is given to the Bagging model for classification. The indexed dataset of the Software Requirement Document is passed as an input to the Bagging classifier, which has two independent classifiers namely, SVM and Decision Tree (DT), to yield the classified documents. From these classified documents, the document has been ranked. These documents are ranked based on the priority of the classification. From the recommended documents, the features are extracted by employing TF-IDF. The recommended documents and ontologies features are passed individually with respect to the classified documents using TF-IDF and semantic similarity pipeline in order to recommend the individual requirements. Semantic similarity is computed between the contents in the individual requirements and ontology as features. If an individual requirement has more than one keyword, then the combined average of the semantic similarity of all keywords is used. The semantic similarity model used is Jaccard similarity and SemantoSim similarity. The Jaccard similarity coefficient is used for measuring the similarity as well as the diversity of the given sample sets. Jaccard similarity is computed using the formula given in Eq. (1). The SemantoSim measure was proposed by Church & Hanks and is represented by Eq. (2).

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{1}$$

$$SemantoSim(x, y) = \frac{p(x, y)log\big[p(x, y)\big] + pmi(x, y)}{log\big[p(y, x)] + [p(x) * p(y)\big]} \tag{2}$$

4 Implementation and Performance Evaluation

The dataset used was Software Requirements Dataset from Kaggle. The implementation is done using Python3.8 as the programming language. The operating system used was Windows 10 and Google Colab environment. The backend is used in MySQL Lite. The processor was an i7 Intel core processor with 32 GB of RAM, and 8 GB of Nvidia graphics card.

The performance of the proposed HSCRD framework is measured by considering Precision, Accuracy, and Recall. Other measures including False Discovery Rate (FDR), F-Measure, and Normalized-Discounted Cumulative Gain (nDCG) are also used. The performance is evaluated for 6141 queries and their ground truth has been collected. The dataset used was Software Requirements Dataset from Kaggle. Standard formulations for Precision, Recall, Accuracy, F-Measure, FDR, and nDCG in terms of a recommendation system were used.

Table 1 presents the performance comparison of the proposed HSCRD approach with other approaches. In order to compare the performance of the proposed HSCRD is

Table 1. Comparison of performance of the proposed HSCRD approach with other approaches.

Search technique	Average precision %	Average recall %	Accuracy %	F-measure	FDR	nDCG
RSRD [7]	86.63	87.69	87.16	87.16	0.14	0.87
Stakeholder + Collaborative Filtering [8]	83.44	86.84	85.14	85.11	0.17	0.82
Content-Based Filtering	84.65	88.25	86.45	86.41	0.16	0.79
Stakeholder Interactions + UML Based Learning	88.16	90.13	89.15	89.13	0.12	0.91
Proposed HSCRD	93.18	95.69	94.43	94.42	0.07	0.96

compared with baseline models namely, RSRD [7], Stakeholder combined with Collaborative Filtering [8], Content-Based Filtering, and Stakeholder Interactions combined with UML Based Learning. The baseline models are evaluated with an exact number of queries in the same environments as the HSCRD approach and the average values of the obtained metrics for 6141 queries for each model are tabulated in Table 1. The RSRD achieves the average Precision of 86.63%, Recall of 87.69%, Accuracy of 87.16%, F-Measure of 87.16%, FDR of 0.14, and nDCG of 0.87. The Stakeholder combined with the Collaborative filtering approach achieves the average Precision of 83.44%, Recall of 86.84%, Accuracy of 85.14%, F-Measure of 85.11%, FDR of 0.17, and nDCG of 0.82. The Content-based filtering approach achieves the average Precision of 84.65%, Recall of 88.25%, Accuracy of 86.45%, F-Measure of 86.41%, FDR of 0.16, and nDCG of 0.79. The Stakeholder Interactions combined with the UML-based learning approach achieves the average Precision of 88.16%, Recall of 90.13%, Accuracy of 89.15%, F-Measure of 89.13%, FDR of 0.12, and nDCG of 0.91. The proposed HSCRD approach achieves the highest average Precision of 93.18%, Recall of 95.69%, Accuracy of 94.43%, F-Measure of 94.42%, a low FDR of 0.07, and a very high nDCG of 0.96.

Figure 3 represents the Precision% vs the Number of Recommendations for each model. The content-based filtering approach does not yield a very high average accuracy mainly because content-based filtering alone is not sufficient for recommending the requirements. Based on the contents of the user query or the requirement domain, it is highly difficult and cumbersome to filter out based on matching keywords and a single strategy namely, content-based filtering only. The approach does not learn any relations, rather it filters directly based on the query or domain of the requirement. Moreover, it fails to unveil micro-requirements and fragmented requirements and it yields a very low nDCG among the baseline models since it does not pave the way to include auxiliary knowledge or background knowledge which leads to low diversity.

Stakeholder Interaction with UML-based learning is quite better than other approaches. This is mainly because stakeholder interactions tend to cover a lot of details that are supplied to the system apart from the query. Detailed stakeholders' interactions mainly highlight the exact needs of the client's requirements. The UML diagrams are divided into several diagrams like class diagrams, sequence diagrams, object diagrams, component diagrams, etc., and when these diagrams are learned, the implementation details and domain-based details also come into the model. As a result, combining both stakeholders' interaction and UML-based learning helps the model to discover the newer requirements along with micro-requirements. But still, there is a scope of improvement of the average precision, accuracy, recall, and F-measure of this model.

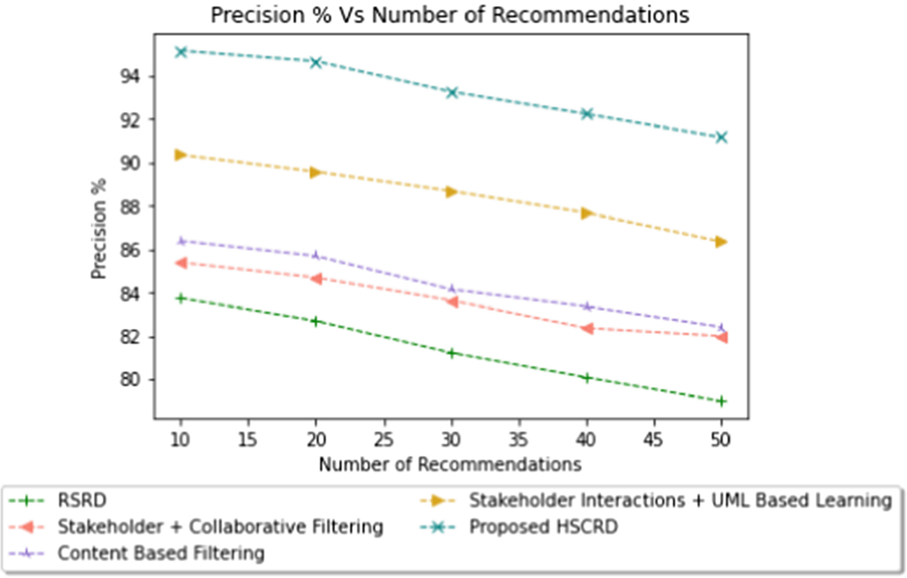

Fig. 3. Precision% Vs number of recommendations

In the Stakeholder with Collaborative filtering approach, stakeholder interaction tends to cover a lot of information apart from the query based on the exact requirements of the clients. In this approach, collaborating filtering is a bad choice because the rating matrix has always been deduced. Deriving this rating matrix or criteria matrix requires ratings based on the stakeholders' interactions for the requirements which is a challenging task and not feasible in practicality.

In the RSRD approach, the binary k-nearest neighbors approach with augmenting profiles of several stakeholders' collaboration are major sources of this approach. They yield less average accuracy because they are relying only on the user profiles which makes the process very tedious and even the difference of opinions among the individual stakeholder users and if it differs in the overall requirement assessment, there is a conflict which happens. The use of only binary KNN is not enough and thus, leads to a need for a much better recommendation approach. Also, the maintenance of a product-by-feature matrix and a feature itemset graph makes it quite tedious and computationally expensive.

The proposed HSCRD approach has a much better performance compared to the baseline models. There are several reasons for that. Firstly, stakeholders' interactions are given very high priority along with several versions of the requirements specification document are included in the environment. In versioning, the evolution from one form of requirement to the final version of requirement is also traced and included for inference in the document by means of TF-IDF. TF-IDF ensures that the documents are ranked initially based on the rarity and frequency of occurrence of terms within the document and across the corpus. Secondly, document indexes are formulated and the ontologies are grown by using standard verified lightweight Upper Domain Ontologies. Fourthly, three distinct knowledge bases are used to incorporate real-world knowledge. Fifth, the Bi-gram and Tri-gram are used in order to increase the lateral density of words. Bagging is chosen with SVM and a highly complex Decision Tree classifier. The computation complexity is balanced and the relevance of the classified documents yielded is much higher. Lastly, the document is ranked and the individual requirements are recommended from the documents by semantic similarity computation and TF-IDF. SemantoSim Similarity has a threshold of 0.75 and Jaccard Similarity has a threshold of 0.5. TF-IDF takes care of macro-requirements from document contents and semantic similarity takes care of the similarity of individual micro-fragmented requirements. Hence, this is a much better approach than the existing approaches.

5 Conclusions

Requirement engineering is one of the important steps in software engineering which can be long and tedious. To improve this process, HSCRD is proposed which is a hybridized semantic approach for knowledge-centric requirement discovery. The stakeholders' interactions are recorded and preprocessed and the individual keywords are input using the TF-IDF model to yield the documents from the Requirement Specification Document Repository. The index words of these documents are extracted and are linked with Upper Domain Ontologies. The ontologies are grown by computing the semantic similarity measures namely, Jaccard similarity and SemantoSim similarity. These grown ontologies are submitted as input to the Wikidata API, Freebase API, and DBPedia API to yield the enriched domain ontologies. The enriched domain ontologies, as features, are passed into Bi-gram and Tri-gram models. Using Bi-gram and Tri-gram of these features, input is given to the Bagging model for classification. Bagging is chosen with SVM and a highly complex Decision Tree classifier. Finally, the recommended documents and ontologies features are passed individually with respect to the classified documents through TF-IDF and semantic similarity pipeline in order to recommend the individual requirements. The proposed HSCRD approach achieves the average Precision of 93.18%, Recall of 95.69%, Accuracy of 94.43%, F-Measure of 94.42%, a low FDR of 0.07, and a very high nDCG of 0.96. The overall accuracy of the proposed approach is much better than the existing approaches.

References

1. AlZu'bi, S., Hawashin, B., EIBes, M., Al-Ayyoub, M.: A novel recommender system based on apriori algorithm for requirements engineering. In: 2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS), pp. 323–327 (2018)
2. Elkamel, A., Gzara, M., Ben-Abdallah, H.: An UML class recommender system for software design. In: 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), pp. 1–8 (2016)
3. Williams, I.: An ontology based collaborative recommender system for security requirements elicitation. In: 2018 IEEE 26th International Requirements Engineering Conference (RE), pp. 448–453 (2018)
4. Avdeenko, T., Pustovalova, N.: The ontology-based approach to support the completeness and consistency of the requirements specification. In: 2015 International Siberian Conference on Control and Communications (SIBCON), pp. 1–4 (2015)
5. Rajagopal, P., Lee, R., Ahlswede, T., Chiang, C.C., Karolak, D.: A new approach for software requirements elicitation. In: Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Network, pp. 32–42 (2005)
6. Shambour, Q.Y., Abu-Alhaj, M.M., Al-Tahrawi, M.M.: A hybrid collaborative filtering recommendation algorithm for requirements elicitation. Int. J. Comput. Appl. Technol. **63**(1–2), 135–146 (2020)
7. Hariri, N., Castro-Herrera, C., Cleland-Huang, J., Mobasher, B.: Recommendation systems in requirements discovery. In: Robillard, M., Maalej, W., Walker, R., Zimmermann, T. (eds.) Recommendation Systems in Software Engineering, pp. 455–476. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-45135-5_17
8. Spertus, E., Sahami, M., Buyukkokten, O.: Evaluating similarity measures: a large-scale study in the Orkut social network. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 678–684 (2005)
9. Deepak, G., Gulzar, Z., Leema, A.A.: An intelligent system for modeling and evaluation of domain ontologies for Crystallography as a prospective domain with a focus on their retrieval. Comput. Electr. Eng., 107604 (2021)
10. Roopak, N., Deepak, G.: OntoKnowNHS: ontology driven knowledge centric novel hybridised semantic scheme for image recommendation using knowledge graph. In: Iberoamerican Knowledge Graphs and Semantic Web Conference, pp. 138–152 (2021)
11. Ojha, R., Deepak, G.: Metadata driven semantically aware medical query expansion. In: Villazón-Terrazas, B., Ortiz-Rodríguez, F., Tiwari, S., Goyal, A., Jabbar, M. (eds.) KGSWC 2021. CCIS, vol 1459, p. 223. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-91305-2_17
12. Yethindra, D.N., Deepak, G.: A Semantic approach for fashion recommendation using logistic regression and ontologies. In: 2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES), pp. 1–6. IEEE (2021)
13. Adithya, V., Deepak, G.: HBlogRec: a hybridized cognitive knowledge scheme for blog recommendation infusing XGBoosting and semantic intelligence. In: 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), pp. 1–6. IEEE (2021)
14. Surya, D., Deepak, G., Santhanavijayan, A. (2021). KSTAR: a knowledge based approach for socially relevant term aggregation for web page recommendation. In: Motahhir, S., Bossoufi, B. (eds.) ICDTA 2021. LNNS, vol. 211, pp. 555–564. Springer, Cham. https://doi.org/10.1007/978-3-030-73882-2_50

15. Krishnan, N., Deepak, G.: Towards a novel framework for trust driven web URL recommendation incorporating semantic alignment and recurrent neural network. In: 2021 7th International Conference on Web Research (ICWR), pp. 232–237. IEEE (2021)
16. Rithish, H., Deepak, G., Santhanavijayan, A.: Automated assessment of question quality on online community forums. In: Motahhir, S., Bossoufi, B. (eds.) ICDTA 2021. LNNS, vol. 211. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-73882-2_72
17. Deepak, G., Kasaraneni, D.: OntoCommerce: an ontology focused semantic framework for personalised product recommendation for user targeted e-commerce. Int. J. Comput. Aided Eng. Technol. **11**(4–5), 449–466 (2019)
18. Deepak, G., Priyadarshini, J.S.: Personalized and Enhanced Hybridized Semantic Algorithm for web image retrieval incorporating ontology classification, strategic query expansion, and content-based analysis. Comput. Electr. Eng. **72**, 14–25 (2018)
19. Deepak, G., Santhanavijayan, A.: UQSCM-RFD: A query–knowledge interfacing approach for diversified query recommendation in semantic search based on river flow dynamics and dynamic user interaction. Neural Comput. Appl., 1–25 (2021)
20. Tiwari, S., Al-Aswadi, F.N., Gaurav, D.: Recent trends in knowledge graphs: theory and practice. Soft Comput. **25**(13), 8337–8355 (2021). https://doi.org/10.1007/s00500-021-05756-8