OBIRS: ONTOLOGY BASED INTELLIGENT RECOMMENDER SYSTEM FOR RELEVANT LITERATURE SELECTION

P. Aruna Saraswathy¹, M. Thangaraj²

¹Research Scholar, Department of Computer Science, Madurai Kamaraj University, Madurai, Tamil Nadu, India.
²Professor & Head Department of Computer Science, Madurai Kamaraj University, Madurai, Tamil Nadu, India.

arunasaraswathyp@gmail.com

https://doi.org/10.26782/jmcms.2020.06.00024

Abstract

Recommender systems are implemented as information filtering agents. In most of the conventional recommender systems, the data about domain is available in limited volumes and suggestions are made to users based on their profile information. This lead to two major problems, insufficient representation of domain knowledge, called 'data sparsity' and lack of user-item interaction, called cold start. These two issues can be addressed with ontology based recommender systems, as they can map domain information with user preferences without losing the semantic richness of the content. This work uses knowledge based method in knowledge aware recommendations to recommend most relevant research papers in digital literature collections. It uses simple methods to construct ontology knowledge graph and uses it for training incremental k-means clustering model. Finally, learning to rank, Adarank algorithm is used to list the top most recommendations for the given user query. The experiments were conducted based on real world unstructured datasets, and results have shown that the proposed model performs well over some of the state-of-the-art baselines.

Keywords: Ontology, NLP, Recommender System, Knowledge Graph, Incremental Learning, Hybrid model, Semantic data model;

I. Introduction

Knowledge bases have three types of knowledge, structural, textual and visual knowledge [XXXIII]. In structural knowledge, the structural components such as entities and their various links are represented. For example, in book recommender system, the various book titles are entities and Author, Publisher, Genre are relations. The inherent network structure implies similarity between various nodes that carry entities [XXVII]. Textual and Visual knowledge explains the structure of text content
and user-friendly visualizations. One kind of such structured information storage is called 'Ontology' [XXXII].

Ontology provides a shared conceptualization and understanding of a domain to facilitate interaction between machines and people [XVI]. Ontology has entities that represent temporal or spatial aspects of real world elements and their respective abstract properties [X]. Ontology is interpreted as knowledge graphs in some scenarios [V]. The difference between the two is that the ontology specifies schema and once the schema is populated with instances, it becomes a knowledge graph.

Knowledge Graphs (KGs) are similar to ontologies that have information about items and the relationship between them. Knowledge graphs have numerous applications in the form of graph completion, word embedding, recommender systems, question-answering systems etc [XXX].

Recommender systems are majorly information filtering agents, therefore, the effectiveness of recommendations is directly proportional to the amount of knowledge it has about the domain. Knowledge based recommender systems are the most effective for their excellent recommendation possibilities. The explainability property of a recommender system is also higher in these kinds of knowledge aware systems, as it includes the historical information of user while making recommendations [IV].

The rest of this paper is organized as follows. Section 2 deals with discussion on related work. In Section 3, the proposed methodology is given. In Section 4, we describe the experimental setup and various evaluation metrics used for validating the system, followed by the discussions in Section 5 and Section 6 concludes the findings.

II. Literature Review

Digitization lead to online hosting of research articles a thriving field [I]. As the number of publications increase exponentially it is next to impossible to manually search related articles of a particular concept. Relying purely based on keyword-based search is not so effective. A recommender system that automatically recommends papers based on the preferences of other researchers with similar interests will be helpful in some areas. However, the number of researchers is not proportionate to number of papers leading to data sparsity issue.

Collaborative filtering is not an effective technique for research paper recommendations [XXIII]. Most of the research paper recommenders are content-based ones that rely on the citations content in each paper. The researcher used subspace clustering based on browsing patterns of users that have similar interests. However, it lacked an efficient ranking mechanism. It also used user rating information that sometimes resulted in cold start issue.

In another work, ontology and the spreading activation model is followed for research paper recommender system. User profile ontology is constructed and combined with spreading activation function. It extracts most influential users in a group and their

Copyright reserved © J. Mech. Cont. & Math. Sci.
P. Aruna Saraswathy et al
interests as the base for making recommendations. It suffered from insufficient data for training the activation function [XXXI].

In one of the literature review articles, the disagreement between choosing content based and collaborative filtering based approach is highlighted [II]. It depended on the application area of the particular recommender. However, recommendations can be improved by using common evaluation frameworks and better user modeling. Lack of wholesome information about the domain is a major drawback in most of the research articles surveyed [VIII]. The user satisfaction was also found to be dependent on diversity of recommendations, serendipity and presentation aspects also.

The aim of most of the recommenders is to provide correct related research articles to the researcher [XXI]. Pairwise similarity calculation is another problem seen in these types of recommenders. To resolve this, ontology based user clustering is undertaken by analyzing semantic similarity based on SVM.

The Problems Found in the Existing Literature to be Addressed by the Proposed System:

- The syntactical retrieval could not analyze the intrinsic preferences of users in information filtering systems
- Keyword search and retrieval suffers from the problem of synonymy, where multiple words will have same meaning and polysemy, where same word will have multiple meanings
- Collaborative models cannot recommend new items to user if they did not come under training set. In other words, new user or new items that have no history of preferences cannot be recommended.
- Content-based models suffer from data sparsity, due to non-availability of complete data about an entity.
- Domain knowledge is under utilized in research article recommender systems.
- Ranking with learning to rank models improve ranking functionality than the conventional similarity scores. It also gives more weightage to pairwise similarity matrix of the system.
- Diversity of recommendations is not properly addressed

III. Proposed Methodology

The proposed methodology is the continuation of our previous work named as 'OBIRS', Ontology Based Intelligent Recommender System [XXXIV]. The recommendations are based on the knowledge obtained through the ontology. The OBIRS architecture as given in Fig.1, works in two phases, one is the knowledge generation phase and the other is the recommender system phase. The various processes ontology engineering, knowledge graph derivation, vector space modeling and incremental ranked recommender model formulation will be discussed in the following subsections. The model is built for recommending the most relevant research articles based on user query. It is to notify the reader about other similar
developments in their preferred area of research and provide a user friendly experience to browse through articles without having to worry about rating them [III]. This work uses semantic knowledge base for knowledge aware recommendations. The domain ontology is built from the unstructured data using concept identification and relation extraction. The 'User Interest Ontology' is built from user behavior with system through log files. The domain and User ontology are mapped together to form the fully represented knowledge base for recommender system. The combined ontology is derived as a knowledge graph. It is followed by computing similarity of nodes using random walk and soft cosine similarity measure to identify contextually similar concepts and represent the knowledge base in vector space model. Incremental clustering is applied to group similar documents together and learning to rank adarank algorithm is used to achieve ranked recommendations.

Fig.1 OBIRS Architecture

III.i. Ontology Engineering

Data related to various fields in computer science was collected from various online sources. The detailed description is given in section 4.1. The first step after data collection is data preprocessing, to remove hyperlinks, html tags, punctuations, special characters, and acronyms. Then the text is segmented into individual sentences. The computer science domain ontology will be constructed using the data corpus. Ontology engineering deals with representing domain knowledge in terms of 'classes', 'attributes' and 'relations' [XXIX]. In the proposed system, the ontology is used to model the hierarchical data abstractions in computer science domain. Ontology population strategies are given in the following sections.
III.i.a. Concept Identification

For modeling unstructured data the latent semantics have to be extracted. The semantic aspects are expressed mostly in the 'subject', 'object' and 'predicates' terms present in the sentences. To identify these terms, our system uses dependency tags and parts-of-speech (POS) Tags. Initially, the corpus is loaded to the 'spacy' NLP text library to assign dependency tags. In subsequent reruns, the POS tags are assigned to the dependency tags that identified major keywords related to the topics, such as machine, data, algorithm etc. Some of the frequent dependency tags obtained are, 'amod', 'compound', 'ROOT', 'dobj', 'pobj', 'nsubjpass', 'csubj', 'conj', etc. The dependency tags are further augmented with their corresponding POS tags such as 'NN', 'NP', 'VERB', 'ADV', 'PROPN' etc. "With the identification and tagging of topic keywords the corresponding relations can be extracted."

III.i.b. Concept Relation Extraction

It has a famous hypothesis that states, "Words with similar meanings occur in similar context". To extract relations, from the preprocessed data, 'Hearst' and 'Regex' patterns are used. The aim is to create taxonomy from a set of computer science-related keyword phrases. The concept identification module completes parts of speech tagging before extracting Hearst patterns [XV]. Most of the relations will take the form such as 'Is-a', 'include', 'and', 'kind-of', etc.

Table 1: Sample Hearst + Regex Patterns for Concept Relation Extraction

| Pattern                                                                 |
|------------------------------------------------------------------------|
| (NP_\w+ (. )?such as (NP_\w+ ?(. )?and |or |?)+)\'first'),            |
| (such NP_\w+ (. )?as (NP_\w+ ?(. )?and |or |?)+)\'first'),            |
| (NP_\w+ ?(. )?+(and |or |?)+(and |or |?)+)\'first'),            |
| (NP_\w+ (. )?include (NP_\w+ ?(. )?and |or |?)+)\'first'),            |
| 'such (NP_\w+ (. )?as (NP_\w+ ?(. )?and |or |?)+)\'first'),            |
| 'example of (NP_\w+ (. )?be (NP_\w+ ?(. )?and |or |?)+)\'first'),            |
| 'example of (NP_\w+ (. )?be (NP_\w+ ?(. )?and |or |?)+)\'first'),            |
| (NP_\w+ (. )?particularly (NP_\w+ ?(. )?and |or |?)+)\'first'),            |

Here, the syntactic elements are used to extract the semantic patterns from the text data. The neighborhood paradigm of the syntactic analysis reflects the semantic meaning of a sentence. With the help of neighborhood paradigm, dependency parsing was traced to identify the subject, object and predicates. In some places, the modifiers and compound words also played a major role in collecting semantically richer content. This was manually verified from the data to provide complete representation of the domain knowledge. The regex grammar rules are constructed for extracting meaningful triples. The dependency parsing is used to extract other relations that were not included in the grammar rule patterns. The sample relation used in the ontology is given in Table 1. In the Table 2 and 3 the sample relations
found in the ontology is given. It is shown that the relation 'show' is highly found in the corpus and the relation 'such as' extract better key words specific to the domain.

Table 2: Sample relation extracted to form the computer science domain

|             |               |
|-------------|---------------|
| show        | 5548          |
| isa         | 4353          |
| prove       | 1629          |
| demonstrate | 1551          |
| give        | 1292          |
| shown       | 1284          |
| present     | 1252          |
| discuss     | 1100          |

Table 3: Statistics of sample relations obtained from the data corpus

III.i.c. Conflict Resolution

Conflict resolution is a process of ontology refinement in other words, ontology ambiguity removal. It is one of the important steps in ontology population. The Reference is seen when a text expression denotes a non-linguistic object in a sentence [VI]. Coreference is the existence of two or more text expressions that denote a same non-linguistic object. The coreference resolution focuses on indentifying and removing multiple references to entities in the ontology to refine the ontology. In the proposed system, the 'properties' of ontology are combined with union and intersection properties to resolve coreference problem. It normalizes the text in such a way by removing redundant mentions in pronouns and assign appropriate entities. The ontology classes are 'sets', that strive to eliminate duplicates as and when the need arises [XXVI]. Anaphoric and cataphoric references are eliminated using
'Neurel Coref' library. A sample of ontology obtained after the procedures is given in Fig.2

Fig.2: sample ontology for the domain computer science

III.i.d. User Interest Ontology

The user interests are gathered from the website log files [XVII]. Initially, the user behavior in simulated environment is monitored. The parameters of concern are, reading history, time spent on a page and click data. Each user will be identified with unique user ids, but profile information for each user will not be obtained explicitly. The behavior analytics is carried out in the log files to filter areas of interests for each user. A search corpus is constructed for every user that contains their frequently searched terms. The terms are assigned broader topics using k-means Clustering algorithm [XI]. The topics and the relevant keywords are assigned an 'is-a' relation and populated in 'User Interest Ontology'. This ontology will be updated whenever a new user enters the system through the incremental recommender data model along with their search terms. The term threshold for a new user is set at 10 in order to eliminate mere browsers from genuine researchers.

III.i.e. Domain Ontology

Domain ontology is obtained after merging user ontology with the domain data. Both the ontologies are merged at the user-id node, based on concept keyword match. The classes related to the computer science domain are populated further with
keywords from the user ontology. This gives a extended knowledge base for the proposed OBIRS. The user profile is also elaborately represented with domain related concepts. For example, if we have the relationship "Machine Learning particularly Predictive Modeling" in our ontology, an individual instance of this relationship looks like

**MachineLearning**  
| particularly | **Predictive Modeling** |

Domain ontology is entirely populated resulting in the knowledge graph with explicit relationships. The knowledge graph eliminates the need for unnecessary table joins. Using this kind of conversion will allow for more flexible data storage and retrieval using machine learning algorithms. It provides constraint free data updations which improve the efficiency of ontology even in shared environments. The statistical data of the constructed ontology are, number of Hyponym classes is 1156, the number of hyponym subclasses is 1203, number of object properties is 63, the number of data properties is 71 and number of individual values is 5031. This schema represents the domain of computer science which can be further indexed and appended with research articles along with the unique article-ids. The structured data obtained from the ontology is the knowledge base of the proposed system. The ontology merging of user and domain ontology is given in Fig. 3. For a sample user. In the Fig.4 the knowledge graph obtained for the relation 'demonstrate' there were 1500 mentions for the relation 'demonstrate' in the ontology.

**Fig.3:** User interest ontology to domain ontology mapping

*Copyright reserved © J. Mech. Cont. & Math. Sci.*

*P. Aruna Saraswathy et al*
III.ii. Vector Space Modeling

Feature engineering is the process of converting nodes, edges and properties into a continuous low dimensional vector space while preserving the basic structure and relationships of the content. The proposed system extracts three types of features such as, node attributes, local structural features and node embeddings. These features represent properties of nodes, neighborhood nodes and context information with fixed-length vectors. This process is called graph embedding. Machine learning models can only accept continuous data, therefore, the discrete graph nodes are embedded to continuous node vectors.

Random walk on a connected graph say, $G = (N, E)$, $|N| = a$ and $|E| = b$. In initial node $n_0$, it selects “at random” an adjacent node $n_1$, and move towards this neighbor. Then again it selects “at random” another neighbor node $n_2$ of $n_1$, and move towards it. The sequence of nodes obtained through the random walk are $n_0, n_1, n_2, \ldots , n_s, \ldots$ this is a simple random walk on $G$. At each step $s$, we allot a random variable $R_s$ that will take values on $N$. Hence, the random sequence $R_0, R_1, R_2, \ldots, R_s, \ldots$ is a discrete random time process defined on the state space $N$. The traverse length is set as 20. The total number of random walk sequences obtained was 15,054. Then soft cosine similarity using levenshtein distance is used to compute semantic similarity between nodes using feature similarity matrix multiplication in the vector space [Wikipedia]. This will extract the weighted sequences called node embeddings. A fixed length vector of 300 represents every node. The condition for a given concept is that, the soft cosine value should be the sum of scores of the keywords present in each concept. It helps to group similar keywords in close proximity.

Fig.4: Knowledge graph generated by the relation "demonstrate"
III.iii. Incremental Ranked Recommender

The feature vectors are provided as training dataset for incremental k-means clustering algorithm to identify topic clusters. The aim is to have multiple clusters that change dynamically to maintain optimal number of cluster centers. The algorithm connects strongly connected subgraphs and assigns them to one topic and the process is repeated for all data points. The model is built to be incremental to accommodate new data by keeping a threshold value in time duration as '15 minutes browsing session per new user' in the system, this differentiates researchers from mere browsers. The user feedback will be updated in the user interest ontology and mapped to domain ontology knowledge graph and the model is executed again. This model is used to cluster new documents to their concept labels along with an index. This is further referenced to the user-id index to form a combined domain-user-preference knowledge base for the recommender system.

Adarank algorithm based Learning To Rank (LTR) aims to solve the ranking problem as a case of supervised machine learning procedure [XVIII]. It provides optimal ranking over a group of items. The significant aspect of LTRs is its ability to give more weightage to relative ordering of items than individual scores of entities. Based on the user-query, the related top ranked documents will be listed using ranking model $f(Q,D)$ Q denotes the user query and D denotes the document. The aim is to use machine learning to build this model automatically.

IV. Experiments Setup

The experiments were conducted in Linux operating system with 8GB RAM. For more RAM requirements, cloud based Jupyter Notebooks for python-3 were used. For ontology creation Protégé editor was used. The knowledge graphs were built using python and neo4j graph database. The unstructured datasets were obtained from various digital libraries for domain ontology population.

IV.i. Dataset

Raw 'abstracts' and 'titles' of research articles related to the computer science domain were collected on topics related to big data, text mining, predictive analytics, wireless sensor networks, artificial intelligence, machine learning, data structures, algorithms, mobile computing, operating systems, internet of things, database management systems and distributed systems using a 'web scraper'. The data were collected from research articles hosted by various academic databases. This forms the data for domain knowledge. In order to avoid the data imbalance problem and to eliminate the need for sampling procedures, for each topic, 6500 research abstracts were collected to construct a dataset with 84500 instances. The statistics of data corpus taken for the study is given in Table 4. From the table 4, it is shown that the average number of sentences per topic category is 50K, which can represent sufficient information about the domain. The user information related to their reading attributes were extracted from the system log files. Having a well constructed user profile favors in the effective functioning of the recommender system [XII]. This work relies on implicit user behavior data such as, reading history, time spent on system and click
data as it provided an unbiased image of a person [XIV]. The user profile was objectively constructed, as a user does not carry any image when browsing the system.

Table 4: Statistics of Data Corpus taken for the study

| Topic Corpus Scraped       | No. of Documents Fetched | Number of Sentences |
|----------------------------|--------------------------|---------------------|
| Big data                   | 6500                     | 55250               |
| Text mining                | 6500                     | 58502               |
| Predictive analytics       | 6500                     | 71524               |
| Wireless sensor            | 6500                     | 47251               |
| Networks                   | 6500                     | 58006               |
| Artificial intelligence    | 6500                     | 65543               |
| Machine learning           | 6500                     | 70548               |
| Data structures            | 6500                     | 55900               |
| Algorithms                 | 6500                     | 54189               |
| Mobile computing           | 6500                     | 62314               |
| Operating systems          | 6500                     | 50147               |
| Internet of things         | 6500                     | 54783               |
| Database management systems| 6500                     | 66854               |
| Distributed systems        | 6500                     | 50875               |

IV.ii. Comparative Models

We compare the proposed OBIRS with the following Ontology-aware baselines that are more or less similar to ours.

**OBPR** - Ontology-Based Personalized Recommender implements domain and user interest ontology and a third reference ontology to represent recommendation dataset. It uses multiple wrappers for query translation and expansion based on the ontology specifications [VII].

**OBFR** - Ontology Based Filtering Recommender implements existing ontology for literary themes to provide fiction content based recommendations. The recommendations are depended on the user ratings to find the nearest neighbor[XXVIII].

**OBRS** - Ontology Based Recommender System for higher education domain to recommend courses and institutions to students based on their interests and preferences. It uses machine-learning algorithms to filter and cluster student profiles and recommends courses based on their needs[XXII].
IV.iii. Evaluation Metrics

Recommender systems are mainly concerned about suggesting most relevant results based on the user query or user preferences. It is denoted by the terminology, 'top-k recommendations', where k denotes whole number. Our system is based on implicit user feedback and provides ranked results. It also employs a machine learning data model to automate data updations and effective ranking of results. Therefore, the efficiency can be tested using three different metrics, Accuracy Metrics, Rank-Aware Metrics and Model based Metrics.

IV.iii.a. Accuracy Metrics

The conventional accuracy metrics such as Precision, Recall and F1-Score are evaluation metrics for document retrieval and binary classification systems. These metrics are improved to suit the needs of the recommender systems. The accuracy of recommendations depends on the 'relevancy' of the retrieved documents. In a narrower sense, the accuracy of top-N recommendations is the exact measurement of recommender accuracy. It is denoted by a letter 'k' which is an integer specified by the user.

**Precision@k:** It is the proportion of documents recommended in the top-k set that are genuinely relevant. Number of documents retrieved for a particular user u is given by, $|Pu| = k$ and the user’s genuine interest truth is set as $Gu$. Precision@$k$ reflects the real intent of a user query. It is calculated as,

$$\text{Precision}@k(u) = \frac{|Pu \cap Gu|}{k} \tag{1} [\text{XIX}]$$

**Recall@k:** It is the proportion of relevant documents recommended in the top-k set to the set of relevant documents. The ratio is obtained by dividing relevant documents by set of user's genuine interest truth. Similar to precision, k is set to 1 if no relevant documents were found for given k. It is calculated by

$$\text{Recall}@k(u) = \frac{|Pu \cap Gu|}{|Gu|} \tag{2} [\text{XIX}]$$

**F1 Score@k:** It is the harmonic mean of the precision and recall values. It compares both precision and recall, since for a given query, precision may be lower and recall may be higher. Therefore, it is difficult to measure the accuracy of a system with precision and recall scores alone. $F1@k$ is derived as,

$$F1@k(u) = \frac{2 \times \text{Precision}@k(u) \times \text{Recall}@k(u)}{\text{Precision}@k(u) + \text{Recall}@k(u)} \tag{3} [\text{XIX}]$$

As per Table 5, the system was tested with some sample queries to measure the accuracy of recommendations with respect to the top-k positions. In the initial run, we set k = 20 and extracted the values for precision, recall and F1 score. The scores obtained for the queries were satisfactory, although not optimal. Recall values required to be improved. Hence, we repeated the experiments keeping k value as 100.
Table 5: Accuracy of OBIRS for various user queries @ k=20

| Queries              | HitRate@20 | Recall@20 | Precision@20 | F1@20  |
|----------------------|------------|-----------|--------------|--------|
| Big data             | 41.67%     | 57.74%    | 41.40%       | 48.22% |
| Text mining          | 48.01%     | 48.27%    | 48.26%       | 48.27% |
| Predictive analytics | 47.63%     | 58.50%    | 41.01%       | 48.22% |
| Wireless sensor      | 41.01%     | 50.89%    | 51.63%       | 51.26% |
| Networks             | 40.77%     | 56.37%    | 57.62%       | 56.99% |
| Artificial intelligence | 49.68% | 55.97%    | 52.71%       | 54.29% |
| Machine learning     | 43.81%     | 48.52%    | 43.92%       | 46.11% |
| Data structures      | 48.18%     | 47.47%    | 42.66%       | 44.94% |
| Algorithms           | 45.32%     | 49.86%    | 45.48%       | 47.57% |
| Mobile computing     | 48.00%     | 48.08%    | 54.90%       | 51.26% |
| Operating systems    | 41.23%     | 40.84%    | 42.65%       | 41.72% |
| Internet of things   | 49.57%     | 46.15%    | 58.01%       | 51.40% |
| Database management systems | 41.15% | 49.96%    | 54.04%       | 51.92% |
| Distributed systems  | 42.30%     | 57.07%    | 58.01%       | 57.54% |

Table 6: Accuracy of OBIRS for various user queries @ k = 100

| Queries              | HitRate@100 | Recall@100 | Precision@100 | F1@100  |
|----------------------|------------|-----------|--------------|--------|
| Big data             | 59.58%     | 51.91%    | 64.38%       | 57.48% |
| Text mining          | 53.64%     | 59.58%    | 69.61%       | 64.21% |
| Predictive analytics | 61.48%     | 59.17%    | 61.17%       | 60.16% |
| Wireless sensor      | 50.20%     | 53.13%    | 64.44%       | 58.24% |
| Networks             | 68.96%     | 56.39%    | 65.40%       | 60.56% |
| Artificial intelligence | 66.97% | 50.40%    | 61.43%       | 55.37% |

Copyright reserved © J. Mech. Cont.& Math. Sci.
P. Aruna Saraswathy et al
It is found from Table 6 that when k value was set to 100, the accuracy values improved for the same set of queries. Recall values also improved and so did the overall accuracy of the system. The obtained scores were optimal for providing personalized recommendations. This implies that for more k values F1 scores will continue to increase. The average accuracy obtained for the proposed system is found to be 76.5%, which is a better score for a recommender system, as the model improves with new data, the accuracy will increase further.

IV.iii.b. Rank Aware Metrics

Rank aware metrics are used to define how early the relevant documents appear in the ranked list. If two lists return relevant documents from two different recommender systems, the first list produced relevant documents at the first and third positions, the other list produced relevant documents at the second last, and third last positions, the former is better. These metrics are better suited for measuring the efficiency of recommender system.

**HitRate@k**: It represents the number of test cases denoted by 'nhit' that have the ground truth document recommendations in the top k positions. It is defined as, HitRate@k = nhit, in our experiment, n = 100 and n = 20 are used for the tests. It is calculated by checking if the required document is predicted for a user in his top-k recommendations. Hit rate values are given in Table 5 and Table 6 and it is found that the hit rate increases when k value is 100.

**Average Precision@k**: It is the measure of average of all the precisions where there is True Positive value for a given query. Precision@j is the value of precision for top j recommendations and relevant@j takes binary values 0 when j\(^{th}\) recommendation is not relevant and 1 when j\(^{th}\) recommendation is relevant. It considers both relevance and relative order of recommendations.

\[
\text{avgpre@k} = \frac{1}{k} \cdot \sum_{j=1}^{k} \text{precision@j} \cdot \text{relevant@j}
\]

| Machine learning | 63.69% | 57.90% | 64.59% | 61.06% |
|------------------|--------|--------|--------|--------|
| Data structures  | 65.43% | 51.19% | 68.89% | 58.74% |
| Algorithms       | 64.79% | 54.02% | 62.17% | 57.81% |
| Mobile computing | 50.98% | 59.24% | 62.13% | 60.65% |
| Operating systems| 65.54% | 58.77% | 61.46% | 60.08% |
| Internet of things| 54.87% | 54.52% | 60.56% | 57.38% |
| Database management systems | 69.40% | 59.89% | 63.23% | 61.51% |
| Distributed systems | 66.29% | 56.72% | 62.62% | 59.52% |

Copyright reserved © J. Mech. Cont. & Math. Sci.
P. Aruna Saraswathy et al
meanAverage Precision@k (mAP): It is calculated by the mean of average precision for all queries in the system. It is calculated by,

$$mAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}$$  \hspace{1cm} (5)

According to the equation, Q is the number of queries in the set and AveP(q) is the average precision for a given query, q.

![Fig.5: Comparison of mean Average Precision Values obtained for baselines and OBIRS](image)

denotes

Mean Reciprocal Rank (MRR): Reciprocal rank for a query is 1/rank, where rank denotes the position occupied by the highest ranked document. When it does not retrieve any document for a query, the reciprocal rank becomes zero. Mean reciprocal rank is the mean of all query reciprocal ranks. The rank of the document j among recommended documents for the query test case q is rank_j, its mean reciprocal rank is given as follows,

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}$$  \hspace{1cm} (6) [XIII]

Copyright reserved © J. Mech. Cont.& Math. Sci.
P. Aruna Saraswathy et al
From the Figs 5 and 6 the various queries were given to the system to analyze the ranking efficiency, the proposed system clearly outperforms other baseline models. The average mAP and MRR values are found to be 77% and 78% respectively. This imply that the documents retrieved in the top-k positions are both relevant and occupied a better position in the rank list.

IV.iii.c. Model Based Metrics

OBIRS follows a hybridization method of knowledge and model based recommender procedure. The performance is also analyzed for the machine learning data model. The model runs on top of Incremental k-means Clustering algorithm. Therefore, the efficiency can be tested using basic metrics for clustering algorithms. One such metric is silhouette analysis [XXV].

Silhouette Analysis: It is used to study the measure of separation between various resulting clusters. It shows the number of clusters and distance between points between each cluster. According to Fig.7, after analyzing the cluster separation for our proposed system it is found that the silhouette score obtained for a sample of 14_clusters is 0.349473. The silhouette value of 0.3 is a better value for k, though not optimal. It means, the sample is at modest distance from the boundary of neighboring cluster and very close to the points in its own cluster. The Fig.7 also conveys a mild
overlapping tendency present in the clusters over other clusters. This could be due to the effect of having data from the same domain.

Fig. 7: Silhouette Analysis for Incremental k-means clustering on sample data for OBIRS

V. Discussions

User query is used to develop a target concept using the keywords. The distance between the query vector and concept vectors in the KG is calculated. Based on the user preference, query relevance in case of existing user, and only based on query relevance for new user, the concept vectors most similar to the target vector are retrieved along with the documents indexed to them. They are ranked by descending order of the similarity scores. The unbiased user profiling helped in generating recommendations as soon as the system goes online. This model will be semantically enriched due to the presence of knowledge base that avoided the cold start issue. The incremental nature of the recommender model and ability of knowledge graphs to identify new concepts and research articles, improved the scalability of the system. One more interesting aspect about OBIRS is its explainability. The recommendations were not only relevant to the user but also explained why it has recommended a particular document. It is mainly due to the model's dependence on the implicit user interactions with the system, particularly, the user reading history. The path from the user history to the target document can also be tracked through the KG.

VI. Conclusion

The idea of generalization is very specific for recommender systems. It can be said that the semantic element between classes and their entities expand the knowledge of the system even for new unseen classes. This is the motivation behind the proposed system, OBIRS. The unstructured data corpus was used to represent the domain of computer science. Since it is a narrower domain, sparsity of data was negligible. Based on the value obtained in each evaluation metrics, the proposed system has outperformed the other almost similar state-of-the-art baseline models. The system was tested using multiple metrics to measure the accuracy, efficiency and
automation capability. Though it obtained a silhouette value of 0.3, it is due to the presence of same-domain-knowledge-base. The approximate average mAP and MRR values obtained for various queries were 77% and 78% which is a significant improvement over the existing systems in terms of ranking efficiency. The overall accuracy score obtained through the Precision, Recall and F1 scores is 76.5%. Hitrate values were lower for lesser k values, however, for k values over 100 it is found to be promising. The proposed ontology-aware recommender system tried to model the domain knowledge as representative as possible with lesser human intervention. In future, ripple networks can be used to obtain user interest information which would further improve the quality of recommendations.

References

I Agarwal, N., Haque, E., Liu, H., & Parsons, L. (2005, October). Research paper recommender systems: A subspace clustering approach. In International Conference on Web-Age Information Management. Springer, Berlin, Heidelberg. (pp. 475-491)

II Beel, J., Gipp, B., Langer, S., & Breitinger, C. (2016). Paper recommender systems: A literature survey. International Journal on Digital Libraries, 17(4), 305-338.

III Bess Schrader (2020). What’s the Difference Between an Ontology and a Knowledge Graph? Global Knowledge & Information Management Services. https://enterprise-knowledge.com/cms/assets/uploads/2020/01/Ontologies-vs-Knowledge-Graphs.pdf.

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

V Cheyer, A. (2018). U.S. Patent No. 10,002,189. Washington, DC: U.S. Patent and Trademark Office.

VI Garanina, N., Sidorova, E., Kononenko, I., & Gorlatch, S. (2017). Using multiple semantic measures for coreference resolution in ontology population. International Journal of Computing, 16(3), 166-176.
VII Ge, J., Chen, Z., Peng, J., & Li, T. (2012, August). An ontology-based method for personalized recommendation. In 2012 IEEE 11th International Conference on Cognitive Informatics and Cognitive Computing (pp. 522-526). IEEE.

VIII George, G., & Lal, A. M. (2019). Review of ontology-based recommender systems in e-learning. Computers & Education, 142, 103642.

IX Giorgos Papachristoudis. (2019). Popular evaluation metrics in recommender systems explained. https://medium.com/qloo/popular-evaluation-metrics-in-recommender-systems-explained-324ff2fb427d.

X Grossmann, R. (2019). The existence of the world: an introduction to ontology. Routledge. 06-Mar-Philosophy - 146 pages.

XI Iannacone, M., Bohn, S., Nakamura, G., Gerth, J., Huffer, K., Bridges, R., ... & Goodall, J. (2015, April). Developing an ontology for cyber security knowledge graphs. In Proceedings of the 10th Annual Cyber and Information Security Research Conference (pp. 1-4).

XII Isinkaye, F. O., Folajimi, Y. O., & Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. Egyptian Informatics Journal, 16(3), 261-273.

XIII James Mishra. (2017). Mean Reciprocal Rank (MRR). https://machinelearning.wtf/terms/mean-reciprocal-rank-mrr/.

XIV Joachims, T., Granka, L., Pan, B., Hembrooke, H., & Gay, G. (2017, August). Accurately interpreting clickthrough data as implicit feedback. In ACM SIGIR Forum (Vol. 51, No. 1, pp. 4-11). New York, NY, USA: Acm.

XV Kawamura, T., Sekine, M., & Matsumura, K. (2017). Detecting Hyponym/Hypernym in Science and Technology Thesaurus Using Entropy-Based Clustering of Word Vectors. International Journal of Semantic Computing, 11(04), 433-449.

XVI Konys, A. (2018). Knowledge systematization for ontology learning methods. Procedia computer science, 126, 2194-2207.

XVII Li, G., & Chen, Q. (2016). Exploiting explicit and implicit feedback for personalized ranking. Mathematical Problems in Engineering.
XVIII Li, H. (2011). A short introduction to learning to rank. *IEICE TRANSACTIONS on Information and Systems, 94*(10), 1854-1862.

XIX Lv, F., Jin, T., Yu, C., Sun, F., Lin, Q., Yang, K., & Ng, W. (2019, November). SDM: Sequential deep matching model for online large-scale recommender system. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (pp. 2635-2643).

XX Munir, K., & Anjum, M. S. (2018). The use of ontologies for effective knowledge modelling and information retrieval. *Applied Computing and Informatics, 14*(2), 116-126.

XXI Neethukrishnan, K. V., & Swaraj, K. P. (2017, February). Ontology based research paper recommendation using personal ontology similarity method. In *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)* (pp. 1-4). IEEE.

XXII Obeid, C., Lahoud, I., El Khoury, H., & Champin, P. A. (2018, April). Ontology-based recommender system in higher education. In *Companion Proceedings of the The Web Conference 2018* (pp. 1031-1034).

XXIII Park, D. H., Kim, H. K., Choi, I. Y., & Kim, J. K. (2012). A literature review and classification of recommender systems research. *Expert systems with applications, 39*(11), 10059-10072.

XXIV Parlak, B., & Uysal, A. K. (2019). On classification of abstracts obtained from medical journals. *Journal of Information Science, 0165551519860982*

XXV Pedregosa et al., (2011). Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830.

XXVI Prokofyev, R., Tonon, A., Luggen, M., Vouillez, L., Difallah, D. E., & Cudré-Mauroux, P. (2015, October). SANAPHOR: Ontology-based coreference resolution. In *International Semantic Web Conference* (pp. 458-473). Springer, Cham.

XXVII Raj (2019). Creating Smart — Knowledge Base Systems (KBS) using advanced NLP library. Towards Data Science. May 4. https://towardsdatascience.com/creating-smart-knowledge-base-systems-kbs-using-advanced-nlp-library-b5c21dfafcd1.

XXVIII Sheridan, P., Onsjö, M., Becerra, C., Jimenez, S., & Dueñas, G. (2019). An Ontology-Based Recommender System with an Application to the Star Trek Television Franchise. *Future Internet, 11*(9), 182.
XXIX Sure, Y., Tempich, C., & Vrandecic, D. (2006). Ontology engineering methodologies. *Semantic Web Technologies: Trends and Research in Ontology-based Systems*, 171-190.

XXX Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., & Guo, M. (2019). Exploring high-order user preference on the knowledge graph for recommender systems. *ACM Transactions on Information Systems (TOIS)*, 37(3), 1-26.

XXXI Weng, S. S., & Chang, H. L. (2008). Using ontology network analysis for research document recommendation. *Expert Systems with Applications*, 34(3), 1857-1869.

XXXII Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, 13(3), 55-75.

XXXIII Zhang, F., Yuan, N. J., Lian, D., Xie, X., & Ma, W. Y. (2016, August). 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).

XXXIV Thangaraj, M. & Aruna Saraswathy, P. (2019, October). Ontology Based Recommender System using Fuzzy Clustering Technique. *International Journal of Engineering and Advanced Technology (IJEAT)*, 9(1), 6412-6418.