Unsupervised Query-based Document Recommendation Using Deep Learning

Wael Alkhatib, Steffen Schnitzer, Tim Steuer, and Christoph Rensing

TU Darmstadt, Communication Multimedia Lab,
Rundeturmstr. 10, 64283 Darmstadt, Germany
{wael.alkhatib,christoph.rensing,steffen.schnitzer,tim.steuer}@kom.tu-darmstadt.de

Abstract. Traditional recommender systems, which aim to recommend documents, rely on statistical information only (collaborative filtering approaches) or on meta-information describing the user profile, the documents as well as their relations. They poorly consider the text semantics and often suffer from homophily by recommending documents which are similar to the documents already acquired by the user. In this work, we propose a unsupervised query-based recommender system for the recommendation of textual documents. The system relies on Deep Semantic Similarity Model (DSSM) to implicitly measure the semantic similarity between the user's interest, represented the user's history or interests as a query of keywords, and the available documents. Our approach uses an automatically generated ontology. This ontology is used to formulate the queries and to interpret the semantic relatedness between user preferences in the user model and the concepts representing the documents. The experimental results show that our system significantly outperforms the baseline in terms of F_{Score}.

Keywords: content-based recommender system; word embbedings; deep learning; DSSM, ontologies.

1 Introduction

The rapid expansion of the deliverable knowledge on the web along with the rise of big data keeps users overwhelmed. This triggered the need for intelligent recommender systems which support personalized decisions through large amounts of available information according to the user's interests and preferences. Recommender systems aim to reduce this burden of information overload by predicting items of interest to a user.

A large number of fundamental works can be found in the area of recommender systems [5]. Many of the prominent approaches are based on collaborative recommenders which use "User Behaviour" for recommending items [11, 23, 26]. This kind of recommender systems suffers from many shortcomings. Firstly, the cold start problem for new users is hard to tackle since no knowledge is available about their preferences. Similarly, new items would not be recommended until rated by a substantial number of users. Additionally, they poorly perform for individuals who do not consistently agree or disagree with any group of people [13]. Finally, there is a lack of transparency, a collaborative system is a black box, which yields a recommendation without reasoning. Therefore there is little confidence in the recommendation.

Content-based filtering is prevalent in Information Retrieval, where the multimedia and text content of documents is used to retrieve documents relevant to a user's query [19]. Content-based recommenders provide recommendations by comparing representations of content describing an item to representations of users's interests. The advantages over collaborative recommender systems are user independence, transparency and better handling of the new-items problem. Transparency can be achieved by a list of content features or descriptions that caused the recommendation. However, content-based techniques are limited by the features that are associated either automatically or manually with the item. Another major challenge is whether the representation captures all the aspects that influence the user's preferences or interests. Similarly to collaborative recommenders, content-based recommenders suffer from the cold start problem for a new user.

Semantic-based recommender systems have been proposed in order to give the recommender system better reasoning ability [31]. However, the need for domain-specific knowledge limits their applicability.

Our method is inspired by the research field of Deep Learning. Recently, Deep Learning has yielded promising results over different "Natural Language Processing" problems, including semantic parsing [34], search query retrieval [28], sentence modeling and classification [14], name tagging and semantic role labeling [4], relation extraction and classification [17]. Also in the area of recommender systems, Deep Learning has been successfully used [32, 37, 3, 22]. Deep Learning is characterized by implicit feature engineering. It is capable of extracting features with better quality than manually hand-crafted features. Microsoft research developed two promising latent semantic models leveraging deep learning techniques in order to measure the similarity between two documents. These two models are Deep Structured Semantic Models (DSSM) [12] and CLSM Convolutional Latent Semantic Model (CLSM) [27]. DSSM was designed for information retrieval and web search ranking, taking advantage of the availability of a huge amount of click-through data. Each data instance holds the web page title, the query and the relevance strength between them. DSSM is a latent semantic model, with a deep neural network, that projects documents and user queries into a common low-dimensional space. Therefore, the relevance of a document given a query is equivalent to the distance between them in that common space.

DSSM can be used in a variety of machine learning tasks such as ranking and classification. We use it to implicitly embed the text semantic in the recommendation process. We propose a query-based recommender system named R-DSSM, which reflects the recommendation as results of a query of the user's preferences taking into consideration his previous knowledge and current interest. The proposed model learns hidden latent features using the DSSM structure to implicitly measure the semantic similarity between a query representing the user's interest and the available documents under unsupervised conditions.

Our contributions can be summarized as follows:

– To the best of our knowledge, R-DSSM is the first content-based recommender system that reflects the user's interests as a query of keywords and relies on a deep model to learn the semantic similarity between the query and the available documents under unsupervised conditions.
– R-DSSM assumes unstructured documents and keyword-based user profile. It minimizes the reliance on explicit representative features of both, the user and the document profiles.
– It alleviates the cold start problem for new users by using a set of keywords as an initial user model. Similarly, the cold start problem for new documents is reduced through the semantic based matching with the user's interests. Additionally, it overcomes the problem of data sparsity by using trigram based word hashing to represent the documents.

The remaining parts of the paper is organized as follows: An overview of related work is provided in Sect. 2. Our methodology to build content-based recommender system is introduced in Sect. 3. Section 4 introduces our experimental settings and evaluation outcomes. Finally, Sect. 5 summarizes the paper and discusses future work.

2 Related Work

With regard to text documents recommendation, we present a variety of methods that directly relate to our approach. These methods fall into two categories. Firstly, ontology and rule-based recommender systems which rely on a predefined user profile, domain ontology and rules as well as rich meta-information about the documents in order to make a recommendation. Secondly, besides traditional recommender systems, we concentrate on approaches that leverage deep learning for recommender systems.

Pujahari and Padmanabhan used decision lists to build a user profile [21]. When a new document arrives, it will be given a rank based on matching against the rules in the decision lists. Then, documents will be recommended based on their rank. Rohani et al. proposed a recommender system based on representing the user's profile as preferences in a hierarchy of categories and documents [24]. The inner nodes hold the categories and sub-categories while the leaves hold the documents. A score is associated with each node to represent the interests of a user in that document or category. To generate recommendations, the system takes the top 10 categories with the highest scores then it searches for documents that are similar to the ones in the top 10 categories.

Yu et al. proposed an ontology-based recommender system for context-aware E-learning [35]. The main constitutive components are three ontologies, namely *Learner Ontology*, *Learning-Content Ontology* and *Domain Ontology*. The *Learner Ontology* represents the learner context e.g., his learning goal, learning interests, location, and the subjects already mastered. The *Learning Content Ontology* defines the properties of contents as well as relationships between them. The *Domain Ontology* is built using existing domain ontologies, also the concepts are ordered in a hierarchy. The system ranks the learning contents according to the learner's goal in the learner's ontology. This ranking is done by projecting the learning goal and the learning contents on the *Domain Ontology* then measuring the distance between each learning content and the learning goal.

Kumaran and Sankar combined collaborative filtering with semantic representation [15]. In their approach, the recommendation process is based on the learner performance and the rating of each topic by other learners. It consists of nodes representing the course topics, arcs representing the relationships between two topics and link labels representing the weight (ranking) between two topics. Another semantic-net is used to represent the user profile. It contains information about performance, knowledge level, etc. The user profile keeps track of the user actions and records them. The recommendation is done based on the weight (ranking) between the topics and the learner performance.

Overall, the main limitation of the above studies is the reliance on predefined ontologies or rules to match a user profile with a document. Moreover, the user profile and the documents assumed the availability of rich meta-information which limits the system scalability and applicability over different domains.

Recently, researchers have proposed deep learning structures for tackling recommendation problems. Some studies integrate deep learning with traditional recommender models [32, 37, 3, 8], while others proposed models which rely solely on deep learning techniques [6, 33, 10]. Gong and Zhang proposed a hash-tag recommendation system that solely relies on a convolutional neural network (CNN) for tags recommendation in microblogs [7]. They transferred the recommendation task into a multi-class classification problem. Xu et al. presented a tag-aware personalized recommender systems which uses a deep semantic similarity based structure in order to compute the similarity between a user and an item, where both the user and the item are represented by tag annotations [33]. Similarly other deep learning models: CNN, MLP, RNN, and DSSM were used [36]. The capability of deep learning to learn the non-linear relationship between users preferences and documents, as well as to learn the representative features of document and a user profile are causing a paradigm shift towards deep learning-based recommendation.

However, the previously studied models which use solely deep learning as a recommender, still rely on labeled data to train a deep semantic model to learn the relationship between documents and users preferences. Collecting such labelled data is time-consuming and tedious. In addition, the lack of transparency while using deep learning models in the context of recommending textual documents hinders the interpretation of why a document is recommended. Finally and most importantly, diversity, novelty, and serendipity in recommendation are not intensively addressed since the main goal of the studied models is to provide similar documents.

In this work, we address the different challenges by introducing R-DSSM: an unsupervised query-based document recommendation using Deep Learning. It assumes unstructured documents without meta-information. R-DSSM reflects the user's interests as a query of keywords based on the previously read documents, which provide transparency and interpretability of the recommendations. The model relies on DSSM to learn the semantic similarity between the query and the available documents alleviating the sparsity problem by representing documents based on their word-hashing trigrams of letters. The DSSM is trained in an unsupervised manner by automatically extracting keywords from the unstructured documents and let the DSSM learn the similarity between the original document and the extracted keywords. This is a key point in the proposed solution which makes it domain independent and excludes the need for a huge amount of click-through data.

3 Methodology

The proposed system constitutes of two main components, namely the *User Model* and the *Query-based Recommender*. In addition, an ontology is used, which is created automatically once, using a chain of natural language processing techniques and external knowledge bases. The user model represents the user's interests as a query of keywords. These concepts are extracted from the documents that the user has read so far. The *Query-based Recommender* uses a DSSM model to learn the semantic similarity between the query representing the user's history and the candidate documents. Finally, it provides ranked list of documents based on the explicit matching between the recommended documents and the generated query terms.

3.1 Word Embeddings Creation

This module provides the distributed representations of words and documents contained in the document corpus using *FastText* and *Doc2Vec* respectively.

Word and document embeddings can form an excellent universal representation to optimize the feature space in non-neural models. Word embeddings are numerical representations of words in a reduced space as a vector of numbers. They can capture the words semantic and context. This means semantically-related words are close in the vector space [1]. For word embeddings, a *FastText* model is trained using the whole document corpus. *FastText* is a form of word embedding capable of learning character n-gram representations in addition to the word itself, thus it can take subword information into account [2].

For document representation, *Doc2Vec* model is trained using the whole document corpus. Similarly, *Doc2Vec* is an unsupervised technique to represent sentences, paragraphs and documents in a reduced features space as a vector of numerical values [16]. Using *Doc2Vec* model, documents with similar context will be close in the vector space.

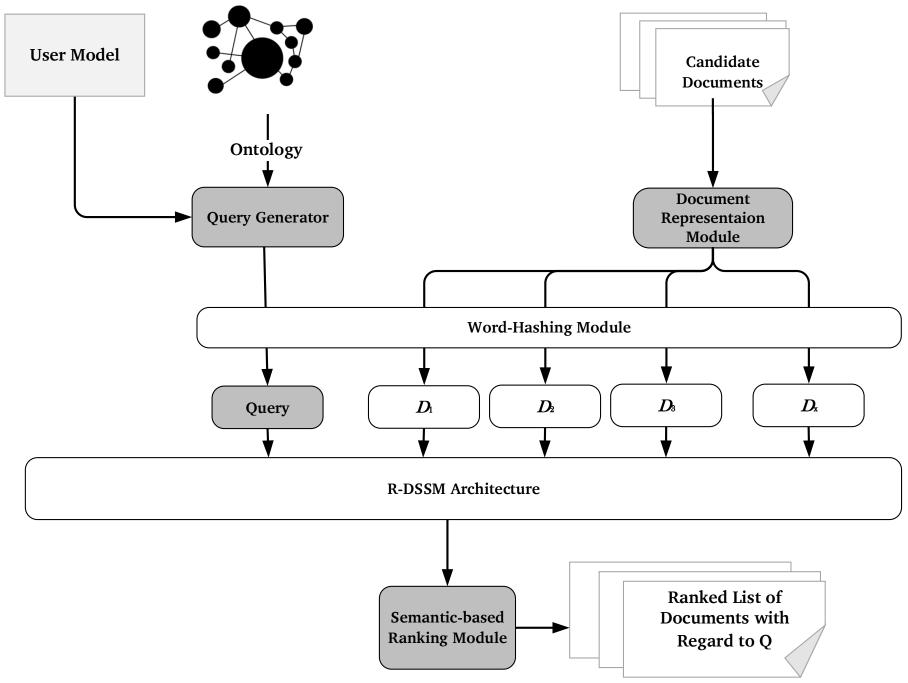

Fig. 1. Block diagram of the query-based recommender.

3.2 Ontology Generation

To build a comprehensive ontology we combine the different lexical databases *WordNet*, *YAGO* and *ConceptNet* to build a basic ontology. *WordNet* [20] is a large semantic network organizing words in synonym sets (synsets). All words and phrases in a synset describe a certain context. Most notably, *WordNet* is an ontology containing different kinds of semantic relations between nouns, namely synonymy, hyponymy, meronymy, antonymy and morphological relations. *YAGO* [18]) is a lexical database which was built by extracting relations from *Wikipedia*, *WordNet*, and *GeoNames*. At the time of writing it contains almost 17 million taxonomic relations as well as various other relation types such as "happened on date" and "lives in". *ConceptNet* [29] provides many non-taxonomic relations such as "has property", "is used for" and "located near". It contains approximately 28 million relations and is available in 304 languages in total. Using the different lexical databases we construct our basic ontology.

Since the lexical databases are static and have small coverage of concepts for particular domains, we expand this initial ontology by extracting additional relations. We crawl new semantic relations from *Wikipedia*. We use a lexico-syntactic pattern-based approach, specifically, we use the six Hearst [9] patterns to detect taxonomic relations. We enrich the first ontology with around 113,000 new taxonomic relations. Ambiguous patterns might return correct as well as incorrect results, for that, we keep the relations found at least 3 times to guarantee a higher precision of the extracted relations. The ontology is further extended using the *FastText* model. We add any concept with a high cosine similarity to a concept already in the ontology as a potential synonym. We set a hard threshold of 0.9 for the cosine similarity which reflects words with similar context or meaning in order to minimize the number of incorrectly selected concepts as synonyms. The threshold is selected based on an ontology enrichment approach using the synonym relation proposed by Alkhatib et al. [1].

3.3 User Model

The user model should provide information about the user's interests and preferences. The user model is built by analyzing the documents in the user's history. Figure 2 illustrates the process of building the user model: First, the noun-phrases (NPs) are extracted from the documents in the user's history using linguistic filters. A combination of 3 linguistic filters is used to extract multi-word NPs from the corpus.

- $Noun\, Noun+$
- $Adj\ Noun+$
- $(Adj|\,Noun)+\ Noun$

Then, we identify the concepts that reflect the interests of a user by applying a corpus-oriented technique i.e. Term Frequency/Inverse Document Frequency (TF-IDF) on these phrases. Noun phrases with high TF-IDF weights indicate concepts which are specific for single or few documents. Terms with low TF-IDF would represent the general interest of the user. The combination of these concepts should be representative of the main topics covered by the user. In the third step, the candidate concepts are checked against our ontology. Concepts which are not found in the ontology or do not have semantic relation with other concepts will be excluded in this phase to filter out any noun phrases that might not present concepts.

The user model should be capable of reflecting the long-term and short-term user's interest. By using the full history during the concept extraction, we can highlight the long-term interest, while using the most recent documents will help identifying the recent short-term interest of a user. In our evaluation scenario, we are considering the full history for building a query from the user model.

3.4 Query-based Recommender System

The proposed recommender system transfers the recommendation process to the information retrieval space. Contrary to traditional recommender with poor usage of semantics, our recommender adapts the state-of-the-art semantic similarity model of DSSM to measure the similarity between the query and the candidate documents. Figure 1 illustrates the process of generating recommendations, which consists of the following four steps:

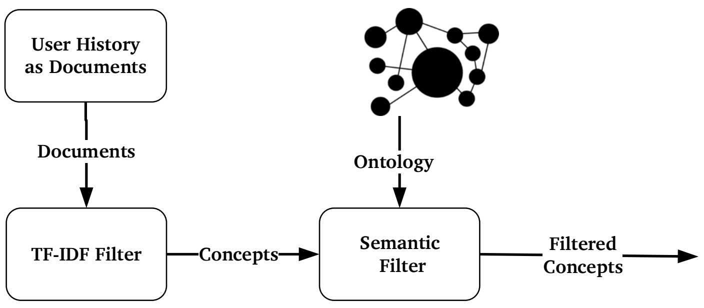

Fig. 2. The workflow of building a user model.

Query Generator The input of this module, is a set of concepts generated by the *User Model*. The *Query Generator* generates a set of queries in such a way that each query partially reflects the user's interests.

The semantic relations between the concepts should be taken into consideration when creating a query. On one hand, the presence of multiple synonyms in the same query could clarify and emphasis the meaning. On the other hand, the presence of a hypernym and its hyponyms i.e. ($Vehivhle \longleftrightarrow Car$) would reduce the query clearness. The *Query Generator* randomly selects a subset of the concepts of the *User Model* and excludes all hypernyms of other concepts in the same subset to form a query.

Document Representation Module This module represents a document as a set of topics. Firstly, the noun phrases are extracted. Then using the *FastText* representation of the noun phrases, a naive approach to cluster the noun phrases is applied: Using a soft cosine similarity threshold of 0.5 between word-pairs, the vectors which are similar to each other are clustered together into one set. At this point, each set is considered to be representing one topic of the document. This was assumed because the vectors in the set are representing all the noun phrases (concepts) that are semantically similar. Topics with only one noun phrase are excluded. By this we filter out all noun phrases that do not represent relevant topics i.e author name, typos, etc..

Deep Semantic Similarity Model (DSSM) Using the *Word-Hashing Module* the query extracted from the user model and the candidate documents are represented as vectors of trigrams of letters. In the last step, the DSSM model is used to obtain a ranked list of the candidate documents for each query. Our proposed model is based-on a modified version of the DSSM structure in [12] as shown in Figure 3. The proposed structure is a fully connected neural network with two non-linear projection layers. The input for the model are the terms of the extended query and the representations of the available documents generated in the previous step. The model has a preprocessing input layer that converts the input queries and documents into Word-Hashing vectors. Word-Hashing is a technique to represent a document as a vector of trigrams. The output of the DSSM is a vector with 120 features that represents the semantics of the input. To measure the similarity, a cosine-similarity layer is attached on top of the DSSM. The output of this layer is a rank for each document regrading the semantic similarity to the query. The negative query-document pairs are generated by matching the original query with a random feature vector as an irrelevant document.

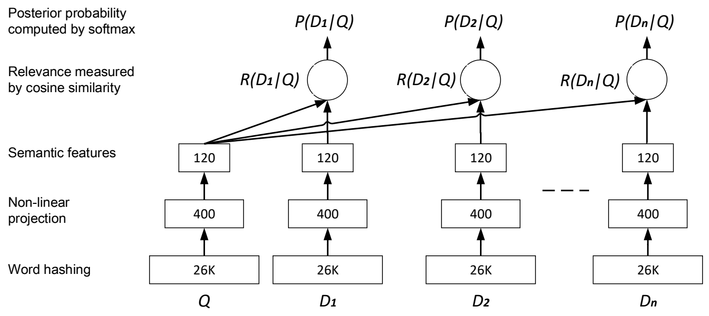

Fig. 3. The proposed DSSM model structure.

Unsupervised Training of the DSSM: The DSSM should be trained with set of documents and their corresponding queries. In order to train the DSSM in an unsupervised way, we automatically extract the keywords representing a document and generate the corresponding queries to train the DSSM on on query-document pairs.

Figure 4 illustrates the workflow of the keywords extraction module. The process starts with extracting a set of keywords using Rapid Automatic Keyword Extraction (RAKE) algorithm, an unsupervised document-oriented keywords extractor [25]. RAKE weights represent the importance of each noun phrase with respect to the whole document. Only keywords which consist of two or more nouns are selected as candidate keywords. Single words might represent very common or high-level concepts which are less probable to represent a document. Using our ontology, we remove all nouns that do not represent concepts such as person name, university name, cities, countries, etc. This is done by matching the extracted keywords with concepts in the ontology. The resulted list might still include concepts that do not represent the main topics of the document. In order to get rid of those concepts, we pass all extracted concepts through word embeddings filter to filter out all concepts that do not have a strong semantic relation with other ones. Using the extracted keywords we generate a set of eight queries for each document.

Then we train the DSSM based on them and their corresponding documents as the positive training examples. The negative query-document pairs are generated by selecting documents with low cosine similarity to the document corresponding to the query using *Doc2Vec* as a document representation.

The number of queries per document was fixed to 8 after an intensive evaluation of their effect on the training loss. A range between 2 and 15 queries was evaluated and the loss showed no significant improvement after using more than eight queries.

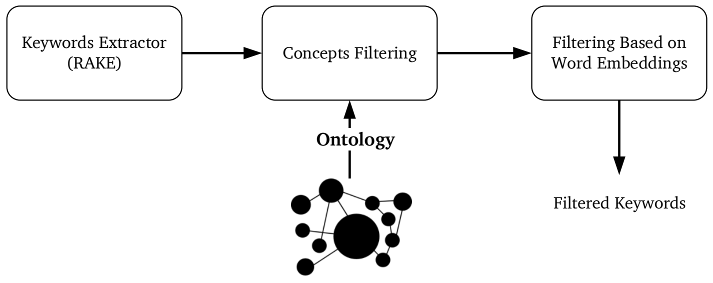

Fig. 4. The workflow of keywords extraction module for DSSM training.

Explicit Semantic-based Ranking Module (ESR) The output of the DSSM is a ranked list of documents. As mentioned, multiple queries are used and thus we have multiple lists of ranked documents. These will be combined based on a majority vote over the different queries. All the documents recommend by only one query will be filtered out. The remaining list might still include documents which are irrelevant or partially related to the basic query. Therefore, the recommended documents are re-ranked based on matching the main keywords in these documents against the set of terms from our basic queries using word embeddings and a hard cosine similarity threshold of 0.9.

4 Evaluation

The experiments carried out aim to study the effectiveness of using R-DSSM for content-based recommendation of documents by analyzing three different aspects, namely the input representation, the network structure and the performance against the baseline. A set of 100,000 ACM papers, published in different ACM conferences [30], was used in order to perform an offline evaluation of the query-based recommender. The research papers have been collected from 51

ACM conferences covering different research areas including *Network Security*, *Software Architecture*, *Database*, *Machine Learning*, *Education* etc. The *Doc2Vec* and *FastText* models used were trained using the *Wikipedia* and the previously mentioned ACM papers.

4.1 R-DSSM Structure Analysis

Tow major aspects were analyzed with regard to the R-DSSM structure, namely the number of hidden non-liner projection layers and the input encoders of the word-hashing trigrams. The R-DSSM structure was optimized by minimizing the cosine-similarity based loss over the training and validation sets. Two different encoders were used to represent the weighted word-hashing trigrams. The first one is Term Count (TC) as a document-oriented technique and the second is TF-IDF as corpus-oriented technique. In all experiments, we use *ReLu* as default activation function for neurons and applied batch normalization. *Adam Optimizer* was used as the optimizer. Additionally, we have used initial learning rate of 0.0001 and trained our model with a batch size of 256 over 150 epochs.

Table 1. Training and validation loss corresponding to the different configurations.

DSSM Layers			**Encoding Using TC**		**Encoding Using TF-IDF**	
L_1	L_2	L_3	**Training Loss**	**Validation Loss**	**Training Loss**	**Validation Loss**
400	120	-	1.0	24.5	0.1	0.6
1000	500	-	0.8	25.0	0.09	2.0
2000	1500	1000	1.5	38.1	0.15	3.1
2000	1500	500	1.3	23.0	0.08	2.7

Table 1 presents the cosine-similarity based loss of the R-DSSM over four different structures and two types of encoders. Using TF-IDF for encoding the weighted word-hashing representation of the query and documents reduces the loss over the training and testing sets compared to using TC. This can be justified by the fact that TF-IDF reflects the importance of the different trigrams for a specific document with regard to the whole corpus. Accordingly, using a more shallow structure of two layers with 400 and 120 neurons respectively performed better compared to more complex structures which essentially related to the amount of training samples that can support optimizing the DSSM as well as the number of trigrams representing the vocabulary. Consequently, in all further experiments the first structure will be used with TF-IDF as the encoding method.

4.2 Detailed Analysis of R-DSSM

In order to analyze the effect of the training set size on the capability of the DSSM to learn a semantic model over the different subject areas, the following experiment has been applied. A subset of 6600 papers was selected to train the

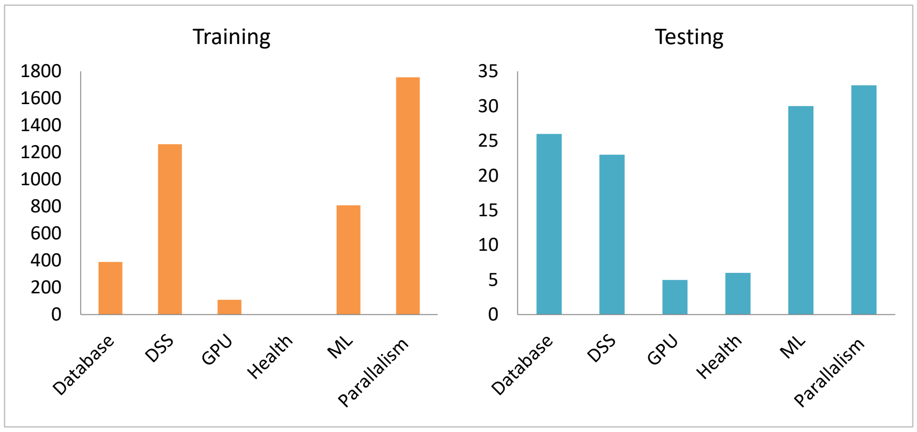

Fig. 5. Distribution of training and testing papers over the different subject areas.

model, while 120 papers were used for testing. Figure. 5 illustrates the distribution of the papers over the different subject areas. The papers in the testing set were manually grouped using their keywords under 8 subject areas. For each paper, the R-DSSM evaluated based on recommending papers covering same subject areas out of the used 6600 for building the model.

Figure. 6 illustrates the precision@k over the different subject areas. The R-DSSM performance for papers with subject area of *Parallelism* is significantly high, while it is lower for papers under other subject areas with less training samples. The performance of the model for *Distributed Software Systems (DSS)* area is significantly low. DSS is at the core of the vast majority of applications ranging from big data centres, embedded systems to networking and cloud systems which requires more data to train a model for learning the semantic similarity under this field.

4.3 Comparison against the Baseline

In order to evaluate the query-based recommender system against the baseline, we used 6600 papers with at least 3 references for each paper from the ACM papers used in the experiments. The recommender system is evaluated by confirming whether the referenced papers are in the top 3 recommended documents. Based on 3-folds cross-validation, we compare our proposed model against two baselines, namely a collaborative recommender and K Nearest Neighbors (KNN) as a recommendation model with *Doc2Vec* for document representation. The KNN model retrieves candidate documents based on the cosine-similarity between the *Doc2Vec* representation of documents.

For the collaborative system, different variations of the user's history were evaluated and the best performance is reported here. The user's history, for both R-DSSM and Doc2Vec+KNN, considered to be the paper to recommend

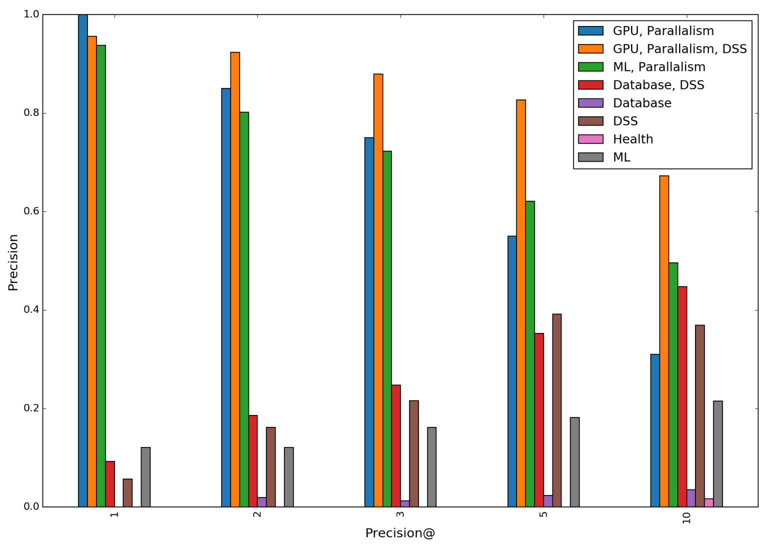

Fig. 6. Precision of R-DSSM over the different subject areas.

its references while for the collaborative system the history are the other papers published by the same author.

Table 2. Comparison of R-DSSM against the baseline.

Recommender System	**Precision**	**Recall**	**F-Score**
The proposed recommender	0.3674	0.4469	0.4033
Doc2Vec+KNN	0.2140	0.1453	0.1731
Collaborative	0.0731	0.0902	0.0808

Table 2 shows a comparison between R-DSSM and the baseline based on the top 3 recommended papers. The R-DSSM overcomes both systems, the Doc2Vec+KNN and the collaborative, with F-Score equal to almost the double of the Doc2Vec+KNN and significantly better compared to the collaborative system in terms to F-Score.

4.4 Explicit Semantic-based Ranking Module (ESR) Effect

We further analyze the impact of explicitly matching the candidate recommendations with the original query in order to re-rank the recommended papers. Table 3 proves that the explicit matching clearly improves the ranking of the

Table 3. Comparison of R-DSSM with and without ERS.

Recommender System	Precision	Recall	F-Score
Without ESR	0.3674	0.4469	0.4033
With ESR	0.6140	0.7473	0.6738

recommended papers, however, it reduces the model ability to provide not just similar papers but also papers with novel knowledge.

5 Conclusion

In this paper we have proposed an unsupervised query-based recommender system. The system relies on DSSM to implicitly measure the semantic relatedness between a query representing the user's preferences and the textual documents available for recommendation. The query-based recommender significantly outperformed the baseline. In addition, the system is featureless and relies on unstructured documents without meta-information. While the proposed model is very general and easy to use it might be better in specific use-cases to augment the user model with additional contextual information. For example, the entire input document or other contextual data, such as the author of a piece of text or the date on which it was written could be used. Other semantic representations of the documents and the queries, instead of word hashing, might improve the model performance. Using the proposed system, novel and personalized recommendation can be provided by manipulating the query extracted from the user model to include more general or specific concepts.

References

1. Alkhatib, W., Herrmann, L.A., Rensing, C.: Onto. kom-towards a minimally supervised ontology learning system based on word embeddings and convolutional neural networks. In: KEOD. pp. 17–26 (2017)
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
3. Chu, W.T., Tsai, Y.L.: A hybrid recommendation system considering visual information for predicting favorite restaurants. World Wide Web **20**(6), 1313–1331 (2017)
4. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. Journal of Machine Learning Research **12**(Aug), 2493–2537 (2011)
5. Drachsler, H., Verbert, K., Santos, O.C., Manouselis, N.: Panorama of recommender systems to support learning. In: Recommender systems handbook, pp. 421–451. Springer (2015)
6. Elkahky, A.M., Song, Y., He, X.: A multi-view deep learning approach for cross domain user modeling in recommendation systems. In: Proceedings of the 24th International Conference on World Wide Web. pp. 278–288. International World Wide Web Conferences Steering Committee (2015)

7. Gong, Y., Zhang, Q.: Hashtag recommendation using attention-based convolutional neural network. In: IJCAI. pp. 2782–2788 (2016)
8. He, R., McAuley, J.: Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In: proceedings of the 25th international conference on world wide web. pp. 507–517. International World Wide Web Conferences Steering Committee (2016)
9. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of the 14th conference on Computational linguistics-Volume 2. pp. 539–545. Association for Computational Linguistics (1992)
10. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939 (2015)
11. Hill, W., Stead, L., Rosenstein, M., Furnas, G.: Recommending and evaluating choices in a virtual community of use. In: Proceedings of the SIGCHI conference on Human factors in computing systems. pp. 194–201. ACM Press/Addison-Wesley Publishing Co. (1995)
12. Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM international conference on Conference on information & knowledge management. pp. 2333–2338. ACM (2013)
13. Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender systems: an introduction. Cambridge University Press (2010)
14. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
15. Kumaran, V.S., Sankar, A.: Recommendation system for adaptive e-learning using semantic net. International Journal of Computer Applications **63**(7) (2013)
16. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning. pp. 1188–1196 (2014)
17. Liu, C., Sun, W., Chao, W., Che, W.: Convolution neural network for relation extraction. In: International Conference on Advanced Data Mining and Applications. pp. 231–242. Springer (2013)
18. Mahdisoltani, F., Biega, J., Suchanek, F.: Yago3: A knowledge base from multilingual wikipedias. In: 7th Biennial Conference on Innovative Data Systems Research. CIDR Conference (2014)
19. Melville, P., Sindhwani, V.: Recommender systems. In: Encyclopedia of machine learning, pp. 829–838. Springer (2011)
20. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
21. Pujahari, A., Padmanabhan, V.: An approach to content based recommender systems using decision list based classification with k-dnf rule set. In: Information Technology (ICIT), 2014 International Conference on. pp. 260–263. IEEE (2014)
22. Rassweiler Filho, R.J., Wehrmann, J., Barros, R.C.: Leveraging deep visual features for content-based movie recommender systems. In: Neural Networks (IJCNN), 2017 International Joint Conference on. pp. 604–611. IEEE (2017)
23. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: an open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM conference on Computer supported cooperative work. pp. 175–186. ACM (1994)
24. Rohani, V.A., Kasirun, Z.M., Ratnavelu, K.: An enhanced content-based recommender system for academic social networks. In: Big Data and Cloud Computing (BdCloud), 2014 IEEE Fourth International Conference on. pp. 424–431. IEEE (2014)

25. Rose, S., Engel, D., Cramer, N., Cowley, W.: Automatic keyword extraction from individual documents. Text Mining: Applications and Theory pp. 1–20 (2010)
26. Shardanand, U., Maes, P.: Social information filtering: algorithms for automating "word of mouth". In: Proceedings of the SIGCHI conference on Human factors in computing systems. pp. 210–217. ACM Press/Addison-Wesley Publishing Co. (1995)
27. Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G.: A latent semantic model with convolutional-pooling structure for information retrieval. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. pp. 101–110. ACM (2014)
28. Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G.: Learning semantic representations using convolutional neural networks for web search. In: Proceedings of the 23rd International Conference on World Wide Web. pp. 373–374. ACM (2014)
29. Speer, R., Havasi, C.: Representing general relational knowledge in conceptnet 5. In: LREC. pp. 3679–3686 (2012)
30. Sugiyama, K., Kan, M.Y.: A comprehensive evaluation of scholarly paper recommendation using potential citation papers. International Journal on Digital Libraries **16**(2), 91–109 (2015)
31. Trewin, S.: Knowledge-based recommender systems. Encyclopedia of library and information science **69**(Supplement 32), 180 (2000)
32. Wang, H., Shi, X., Yeung, D.Y.: Relational stacked denoising autoencoder for tag recommendation. In: AAAI. pp. 3052–3058 (2015)
33. Xu, Z., Chen, C., Lukasiewicz, T., Miao, Y., Meng, X.: Tag-aware personalized recommendation using a deep-semantic similarity model with negative sampling. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. pp. 1921–1924. ACM (2016)
34. Yih, W.t., He, X., Meek, C.: Semantic parsing for single-relation question answering. In: ACL (2). pp. 643–648. Citeseer (2014)
35. Yu, Z., Nakamura, Y., Jang, S., Kajita, S., Mase, K.: Ontology-based semantic recommendation for context-aware e-learning. Ubiquitous Intelligence and Computing pp. 898–907 (2007)
36. Zhang, S., Yao, L., Sun, A.: Deep learning based recommender system: A survey and new perspectives. arXiv preprint arXiv:1707.07435 (2017)
37. Zhuang, F., Zhang, Z., Qian, M., Shi, C., Xie, X., He, Q.: Representation learning via dual-autoencoder for recommendation. Neural Networks **90**, 83–89 (2017)