IADIS International Journal on Computer Science and Information Systems
Vol. 19, No. 1, pp. 111-124
ISSN: 1646-3692

BUILDING KNOWLEDGE GRAPHS SUITABLE FOR KNOWLEDGE RECOMMENDATION: EXPERIENCE FROM SHIPBUILDING INDUSTRY

Bo Song[1], Wanting Ma[2], Zuhua Jiang[3] and Ping Fang[2]
[1]China Institute of FTZ Supply Chain, Shanghai Maritime University, Shanghai, China
[2]Logistics Engineering College, Shanghai Maritime University, Shanghai, China
[3]School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China

ABSTRACT

Knowledge graphs have been widely used in recent years to build recommendation systems. However, in related research, the main focus has been on improving recommendation algorithms, with less attention paid to the type of knowledge graph that is more conducive to knowledge recommendations. This paper addresses knowledge recommendation systems in the shipbuilding domain by constructing knowledge graphs in two ways and comparing their performance in knowledge recommendation. One type of knowledge graph is built based on classification tags of knowledge documents, characterized by its simplicity and sparsity; the other is constructed automatically using machine learning, linking knowledge documents together based on concepts and relationships extracted by the algorithm, featuring complexity and density. To recommend shipbuilding knowledge, a context-aware mechanism was employed, gathering information from the user's task environment and linking it to the knowledge graph. Then, using RippleNet, the system spreads the user's interests within the knowledge graph and infers the required knowledge documents. Experimental results show that the sparse knowledge graph achieved better recommendation results. We believe this is due to the human expert experience relied upon during the construction of the sparse knowledge graph, namely a knowledge classification system oriented towards knowledge applications.

KEYWORDS

Knowledge Recommendation, Knowledge Graph, Shipbuilding, Context-Aware, RippleNet

1. INTRODUCTION

People processing creative tasks must seek knowledge from various sources and properly organize them for retrieving and reasoning. To save the time and effort consumed in knowledge seeking, knowledge recommendation is proposed for proactively providing people with useful knowledge (Ji et al., 2023). The research on knowledge recommendation can benefit from the development of general recommendation techniques such as collaborative filtering and content-based filtering, nevertheless, it has unique traits that are determined by the nature of knowledge-intensive tasks and application domains. A common agreement on applying information item recommendation in a complex environment is that the context of user interests is of great importance, and context-aware recommendation has become a trend (Zammali and Yahia, 2021; Sun et al., 2021). Although context awareness has been well studied in recommending items such as movies and restaurants, the context information used in these daily life recommendation situations mainly consists of time and location (Baltrunas et al., 2012), which is much simpler than the knowledge context used for recommending task-relevant knowledge (Song et al., 2016).

Nowadays a lot of recommender systems are based on collaborative filtering or its variants, as these techniques have high accuracy and diversity in recommendation performance. However, there are two well-known problems perplexing this kind of recommender systems: the sparsity of user-item interaction data and the cold start problem (Guo et al., 2020). The two problems become even more serious when considering context, as the user-item interaction in a certain context is much scarcer than that in a general situation. In order to establish some relations between users and items which are not connected in the beginning, knowledge graph (KG) comes into play. KG extents the concept of semantic network and is capable of modelling large-scale relational data with easy-to-use databases and software. When used in recommender systems, KG is usually designed in such a way that the nodes representing users are connected to the nodes representing items to be recommended via the "interact with" relation, and the user/item nodes are interconnected with each other via shared properties (Kaur et al., 2023). Leveraging graph-based machine learning algorithms, a recommender system can tactfully propagate user interests to the items that have not been browsed by the user yet, so as to solve the aforementioned data sparsity and cold start problems and improve the recommendation performance (Wang et al., 2018).

Recent research in recommendation systems has seen a surge in incorporating KGs to enhance performance and explainability. For example, Ma et al. (2023) propose a novel method that integrates KGs with graph convolution networks (GCN) to alleviate error propagation and improve recommendation diversity and relevance. Chang et al. (2023) introduce MKCGN, a meta-relation-assisted graph neural network to address data sparsity issues and optimize user/item representation quality for better recommendation performance. The AGRE recommendation algorithm codes the paths between users and items with a specified RNN to accurately learn the user's preferences (Zhao et al., 2023). On the other hand, Cai et al. (2022) explore explainable recommendation using a knowledge graph and evolutionary algorithm, achieving a balance between precision, diversity, novelty, and explainability. The KGIN model learns the intents behind user-item interactions with KG, offering interpretable recommendation and significant performance improvements (Wang et al., 2021).

In the realm of KG construction, the integration of Named Entity Recognition (NER), Relation Extraction (RE), and Entity Linking (EL) forms the classical approach to building knowledge graphs from textual data. This methodology involves first identifying and classifying entities within the text (NER), then determining the relationships between these entities (RE), and finally linking these entities to unique identifiers within a knowledge base (EL) (Kejriwal, 2022). While NER and RE are traditional NLP tasks, EL is specific to KG and is similar to word disambiguation in nature. For example, Li et al. (2022) explores enhancements in entity linking by incorporating structural information from knowledge graphs to better manage entity disambiguation and linking. Recently, Large Language Model (LLM) begins to play a role in KG construction (Vizcarra et al., 2024), which eases the generation of KG elements with the rich knowledge learned in the pretraining of LLM. Despite all the automated construction methods, constructing robust KGs remains challenging due to issues like the need for extensive domain expertise, the complexity of data integration, and the scarcity of high-quality annotated datasets (Kejriwal, 2022).

Based on current research findings, achieving knowledge recommendation that better meets user needs entails two primary considerations. First, a finer granularity of user interests must be mined from task information, typically achieved through context awareness. Second, establishing more connections between users and knowledge items is necessary to address data sparsity and cold start issues. However, in engineering domains, a significant challenge lies in the costs associated with establishing these data connections. Within enterprises, directly accessible user/item relations are often sparse and established via mature business processes in necessary scenarios, whose expansion requires manual intervention of domain experts. To establish rich connections between users and items at low cost, one can use concept mining and relation mining techniques to automatically construct a knowledge graph encompassing arbitrary granularity of concepts and relations. To the best of our knowledge, there has been few research evaluating the impact of different knowledge graph structures on knowledge recommendation performances. This paper proposes the Context-aware Knowledge Graph-based Recommendation Network (CKGRN), which is tested in a shipbuilding enterprise with two types of KGs: one derived from enterprise resource associations and the other obtained via information extraction algorithms. By leveraging user browsing records as input, CKGRN can propagate user interests in the KG and provide personalized knowledge recommendations. Comparative experiments will show which is the better way of building a KG for inner enterprise knowledge recommendation.

2. METHOD

In this paper, we propose two methods of building KGs in a shipbuilding enterprise, and then use the constructed KGs in a Context-aware Knowledge Graph-based Recommendation Network (CKGRN) model, to see which method is better. CKGRN is composed of three parts. The first part is a KG relating different knowledge items, users and concepts, which is the foundation of the KG-based recommendation engine adopted by CKGRN. The second part is an interest model that assigns interest score to KG nodes and lets the score decay with time. The interest model enables CKGRN to track a user's task processing steps and keep the right information to reason with. The third part of CKGRN is a context-aware recommendation engine, which is built by adding a context-aware mechanism to a KG-based recommender. The context-aware mechanism works in such a way that it actively links entities in a user's working environment to nodes defined in the KG, and then treats the linked nodes as browsed items. The overall framework of CKGRN is shown in Figure 1, and each part of it will be introduced in following subsections.

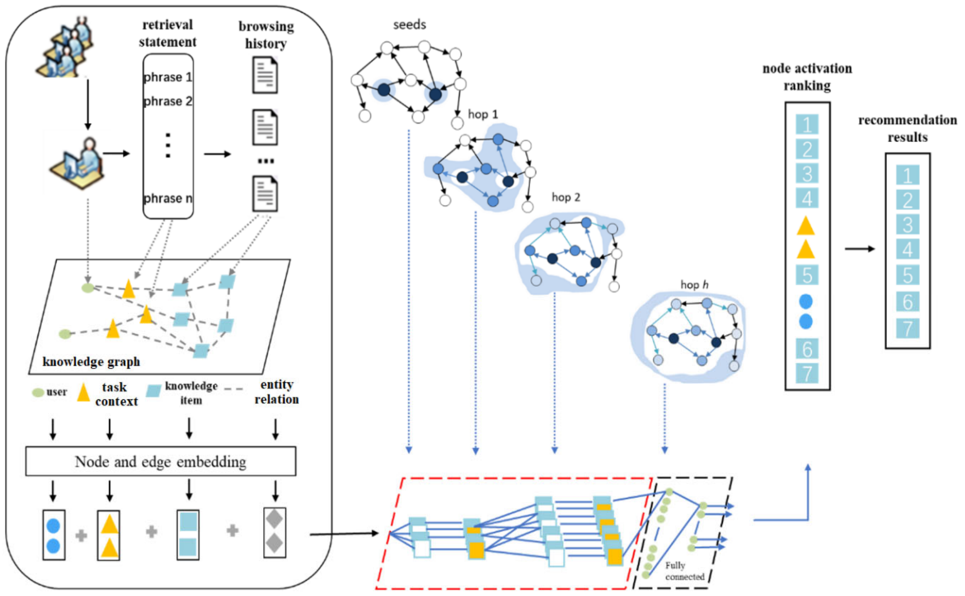

Figure 1. The overall framework of CKGRN

2.1 Knowledge Graphs Supporting Recommendation

In KG-based recommender systems, KGs play a crucial role by enhancing the recommendation logic with rich, structured semantic relationships among items. They allow recommender systems to utilize connections and attributes of items, such as genres, authors, and contextual information, to provide more accurate, relevant, and personalized recommendations. KG-based recommendations can also alleviate a significant issue of missing user-item interaction records. This is achieved by establishing connections between users and items that lack direct interactions, through multi-hop associations of attributes, thus enabling data augmentation. For the shipbuilding scenario considered in this paper, building a recommender upon KG is particularly crucial, as users tend to solve problems with their existing knowledge and rarely browse knowledge documents, leading to sparse interaction records with these documents.

Previous recommendation studies based on KG usually adopt publicly available KGs or build a single domain-specific KG for use. Due to the cost of building a KG from the beginning, few researchers would construct several KGs with the same topic to test their performances, thus leaving a research gap about how KGs with different structures affect recommendation outcomes. In this paper, we will construct KGs using two representative approaches, and obtain two KGs with different structures. The most significant difference between the two KGs is the number of nodes and relations representing the same quantity of knowledge items, which can differ more than 10 times. We refer to the KG with fewer nodes and relations as the sparse KG and the one with more nodes and relations as the dense KG.

2.1.1 Sparse Knowledge Graph

The sparse KG is constructed from the meta information associated with each of the knowledge items in the studied shipbuilding enterprise. An example of a knowledge item is shown in Table 1. There are 7 types of information comprising a knowledge item: *ship type, ship part, keyword, case type, case name, analysis* and *solution*. While *case name, analysis* and *solution* are the main content of a knowledge item, *ship type, ship part, keyword* and *case type* are the meta information used for classifying, managing and retrieving knowledge items. A sparse KG can be constructed by adding *hasShipType, hasShipPart, hasKeyword* and *hasCaseType* relations between each knowledge item and its four meta information, as shown in Figure 2. It can be seen that the smallest knowledge unit in the sparse KG is a knowledge item. One cannot know the detail of a knowledge item by looking at the KG and reasoning with attributes of a knowledge item is not applicable. Despite the coarse granularity of knowledge represented in the sparse KG, all the nodes and relations in the sparse KG are in fact created by experts in the enterprise, and no noise information has been introduced into the sparse KG during its construction process.

Table 1. A knowledge item

Ship type	400K VLOC	Ship part	Hull
Keyword	Web length	Case type	Drawing
Case name	Layout T-row web length error		
Analysis	Ship H1XXX in section 124 layout T-row web length is not correct, 20mm short. Some designers lack experience. A T-row is embedded on site, waste production hours.		
Solution	For T-rows with truncated ends, attentions should be paid to the concept of length correction, and each group leader needs to do spot check more in the process of drawing and proofreading.		

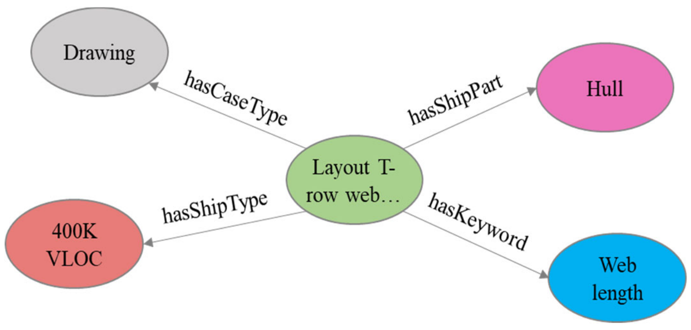

Figure 2. A knowledge item in the sparse KG

2.1.2 Dense Knowledge Graph

The dense KG is constructed by extracting term-level semantic relations between knowledge items and inside a knowledge item. The primary goal of constructing such a dense KG is to make knowledge items reasonable. For example, if we extract a series of (*Object, Attribute, Value, Unit*) tuples such as "web thickness xxx mm" and "deck length xxx m" from the knowledge items, then it will be possible to compare these features across the knowledge items and enable retrieval statements like "find web problems in ship hull with deck longer than 150m". In this paper the dense KG is used for knowledge recommendation instead of knowledge reasoning, as the dense relations among knowledge items provide a lot of paths for the user interest to spread.

The dense KG is constructed in the following way. First, we annotate some knowledge items with the entity types and relation types listed in Table 2, then we use the annotated data to train a BERT-CRF model for named entity recognition and a BERT-LSTM model for relation extraction, at last, the trained models are used to annotate the remaining knowledge items automatically. While in the sparse KG a knowledge item usually has 4 relations with other concepts, the number of concepts related with a knowledge item in the dense KG is determined by how many *Objects* are mined from the item and thus can reach a large value. Figure 3 is a visual comparison of the sparse KG and the dense KG, from which we can see that for the same number of knowledge items, the dense KG has far more nodes and relations than the sparse KG.

In previous recommendation studies based on KGs, there has been few considering the impact of different KG structure. Although some studies have tested their proposed methods on different KGs with different number of nodes and relations, these KGs are of different topics and the test datasets are also different. In this study the sparse KG and dense KG are constructed from the same set of knowledge items and the recommendation scenario is the same either. For the sparse KG and dense KG, the propagation path of user interests will be affected. For example, in ship design, when users search for the keyword "ballast", the interest propagation path in the sparse KG will clearly point to the knowledge items containing the "ballast" keyword and then other knowledge items of the same type; in the dense KG, the interest propagation path will point to nodes representing terms cooccurring with "ballast" in a knowledge item as well as knowledge items containing "ballast", so that the paths reaching other knowledge items will grow in number and length at the same time.

Table 2. Entities and relations defined for mining a dense KG

Entity	Examples	Relation	Explanation
Object	deck, web	RL-of-O	Object O is the head of relation RL
Property	length, thickness	O-of-RL	Object O is the tail of relation RL
Value	200, 9.5	P-of-O	P is the property of object O
Unit	m, mm	V-of-P	V is the value of property P
Action	truncate, embed	U-of-V	U is the unit of value V
Relation	contain, larger than	RF-of-O	RF is the reference of object O
Reference	section 4	A-of-O	O is the actor of action A
		O-of-A	O is the subject of action A
		O-of-O	Object O is related to another object

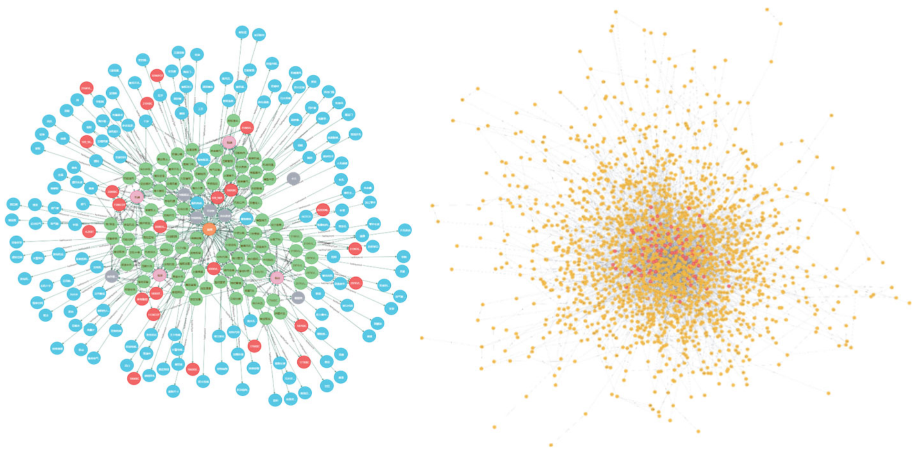
Figure 3. Sparse KG (left) and dense KG (right)

2.2 Interest Model

For user interest modelling, a common assumption is that a user's interest in an item is positively related with the number of clicks and time spent on the item. In a task-focused environment such as shipbuilding, people's interests, or knowledge needs, are driven by the dynamic task context more than their long-standing hobbies. In order to reflect the influence of dynamic task context on knowledge needs, we apply a decay factor to each interest score considering the time passed since the click/browse takes place. Specifically, we adopt the interest model proposed by Wang et al. (2022) to calculate the interest score that a knowledge item or KG node receives.

$$F(U,I)=\begin{cases}0 & count\left(R_L(U),I\right)=0\\ 0 & count\left(R_L(U),I\right)=1,\Delta T\le 30\text{s}\\ \sum_{k=1}^{L}\frac{1}{n-k+1}\cdot\frac{\Delta T^k}{|I^k|}\cdot\delta\left(I^k,I\right) & \text{other}\end{cases}$$
$$\delta\left(I^k,I\right)=\begin{cases}1 & I^k=I\\ 0 & I^k\ne I\end{cases}\tag{1}$$

In Equation 1, $F(U, I)$ denotes the interest of user U in knowledge item I, $R_L(U) = \{(I^1, \Delta T^1), (I^2, \Delta T^2),\ldots, (I^L, \Delta T^L)\}$ is the sequence of browsed knowledge items and their browse time, $1/(n\text{-}k+1)$ is the interest decay factor, $|I^k|$ is the text length of the kth browsed knowledge item, and $count(R_L(U), I)$ is the number of times that knowledge item I has been repeatedly browsed by user U. Knowledge items that have not been browsed are irrelevant to the user and have an interest score of 0. Knowledge items that have been browsed once but for a period shorter than 30 seconds also have an interest score of 0. The remaining knowledge items have an associated interest score calculated from the accumulated browse time coordinated with the text length and decay factor.

2.3 Context-Aware Recommendation Engine

CKGRN has a context-aware recommendation engine, which is built upon an existing KG-based recommender such as RippleNet. Three modifications are made to the original recommender, including context capturing, entity linking and user interest input. The following is a detailed explanation.

(1) Context capturing. In a task-focused environment, if a user clicks on a knowledge item and starts browsing, it means the user has already figured out what knowledge he/she needs. A helpful knowledge recommender in this situation should act in advance, by capturing signals from the user's task context and inferring the user's knowledge need. To do so, we monitor a user's desktop browser and capture the title of the active page as a context message every second. If a context message is the same as the previous one, then the knowledge item or KG nodes corresponding to the message will have duration time +1.

(2) Entity linking. We use a simple entity linking method to map the words in a context message to the nodes in KGs. First, we check for each KG node whether the Chinese character sequence of node name is contained in the context message. If we get yes for a node representing a knowledge item, then the knowledge item is linked to the context message. If we get yes for a node representing a concept, e.g., ship type, ship part, keyword, case type and object, then we check whether at least one neighbor of the node appears in the context message, and if yes, the node is linked to the context message.

(3) User interest input. The original RippleNet uses a 0-1 rating matrix as input and does not consider the decay of user interest over time. In this paper we have the interest model described in section 2.2. The interest model does not affect the training phase of RippleNet, but during the predicting phase, the heads of KG triples in the hop 1 ripple set are multiplied by their according interest score before going to the softmax operation. In doing so the influence of interest scores is passed to the tails of KG triples in the hop 1 ripple set as well as the following ripple sets.

2.3.1 The RippleNet Recommender

The RippleNet (Wang et al, 2018) model is constructed and trained through a sequence of steps that leverage the structure of a KG to propagate user preferences and predict user interest in items. For a given user - target item pair, the user's historically interacted items are treated as seeds within the KG. Starting from the seed set, the model extends the user's interests along the KG links to form multiple ripple sets. Each ripple set contains nodes that are k-hops away from the seed set. RippleNet uses these ripple sets to capture the indirect preferences of the user over the KG. The nodes in the ripple sets interact with the target item embedding to determine how the user's propagated preferences align with the characteristics of the target item. The outputs from the ripple set interactions are aggregated to form a comprehensive user preference profile, which is then used to compute the probability of the user interacting with the target item. The overall framework of RippleNet is shown in Figure 4.

In RippleNet, each node/link in the KG is represented by an embedding, which is a dense vector encapsulating its characteristics derived from the structure of the KG and calibrated by the training user - item samples. RippleNet is trained by optimizing the maximum likelihood of the observed user-item interactions, meanwhile the embeddings in the model are iteratively updated. When doing recommendation, for each item under consideration for recommendation, RippleNet computes interactions between the item's embedding and the embeddings of nodes within the ripple sets, and then selects k items with highest predicted click probability to recommend to the user.

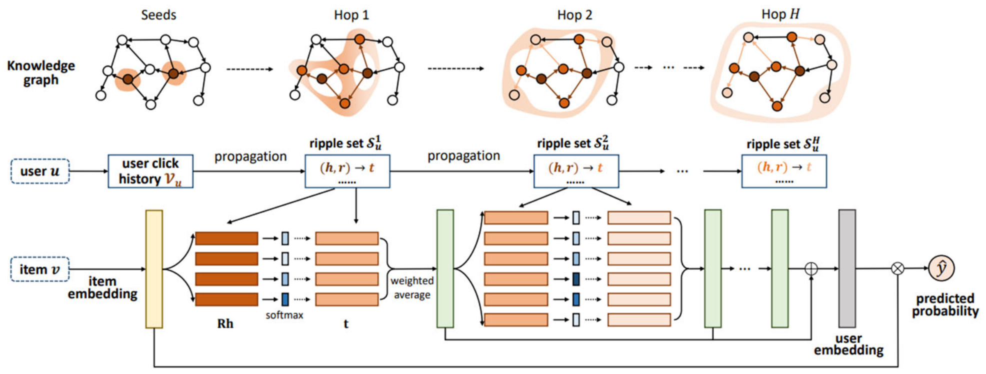

Figure 4. The overall framework of RippleNet (source: Wang et al, 2018)

2.3.2 Other Suitable Recommenders

Generally speaking, all KG-based recommenders can be adapted to construct the context-aware knowledge recommendation model described in this paper. For example, the Multi-task Knowledge-enhanced Representation (MKR) method (Wang et al., 2019), integrates KG embeddings into recommendation systems through a multi-task learning approach. This method leverages a deep learning framework where two main tasks, recommendation and knowledge graph embedding, mutually enhance each other. The connection between these tasks is facilitated by a novel component known as the "cross&compress unit". The cross&compress unit in MKR is designed to automatically control the transfer and interaction of features between the tasks. It does this by modeling high-order feature interactions between items in the recommendation system and entities in the knowledge graph. This dual-task strategy helps to enhance the model's performance, especially in handling the sparsity of user-item interactions by exploiting the rich information embedded in the knowledge graph. Later Gao et al. (2023) introduced the Enhanced Multi-Task Learning and Knowledge Graph-Based Recommender System (EMKR), which built upon the traditional MKR by addressing issues of data utilization and adaptation to different data sparsity levels.

It should be noted that this paper does not primarily focus on the methodology of KG-based recommendation but is concerned with how the structure and quality of KG affect the results of recommendation. Therefore, among the many KG-based recommenders, we only select a few representative ones for experimentation to verify the general influence stemming from the knowledge graph.

3. EXPERIMENT

The experiment with CKGRN is conducted in a shipbuilding enterprise in Shanghai. We collected 90 knowledge items from the production design department of the shipbuilding enterprise. While the sparse KG constructed from the knowledge items has 270 nodes and 499 relations, the dense KG has 3017 nodes and 6038 relations. The personal attributes, retrieval statements, and browsing records of 5 users from the production design department are collected for training and testing CKGRN. In order to avoid possible bias of RippleNet on sparse and dense KGs, we use another KG-based recommender, MKR (Wang et al., 2019) to evaluate CKGRN. The hyper-parameters of the two recommenders are set as Table 3 in the experiment.

Table 3. Hyper-parameters of RippleNet and MKR

Recommender	Parameter	Explanation	Value
RippleNet	dim	KG embedding dimension	16
	n_hop	max hop	3
	kge_weight	weight of KG embedding loss	0.01
	12_weight	weight of l2 regularization in loss	1e-7
	lr	learning rate	0.02
	n_memory	ripple set size for each hop	32
	batch_size	batch size	24
	n_epoch	epoch number	10
MKR	dim	KG embedding dimension	8
	L	number of low layers	1
	H	number of high layers	1
	12_weight	weight of l2 regularization in loss	1e-6
	lr_rs	learning rate of recommendation task	0.02
	lr_kge	learning rate of KG embedding task	0.01
	kge_interval	interval of KG embedding training	3
	batch_size	batch size	24
	n_epoch	epoch number	20

The experimental scenario is click-through rate (CTR) prediction, and we use AUC and ACC to evaluate the performance of different models. ACC reflects the overall recommendation accuracy. Given that the experiment dataset comprises well-curated, typical cases with high-quality features, we assume there is a positive correlation between AUC and recall. This allows AUC to measure both the model's ranking capability and the diversity of its recommendations. The performances of RippleNet and MKR with and without the context-aware mechanism and the interest model are assessed respectively. The experiment results are presented in Table 4. It can be seen that RippleNet outperforms MKR in recommendation effectiveness, and the results obtained with context awareness and interest model are better than those without the two settings. To summarize, in the studied shipbuilding case, the proposed CKGRN model achieves 11.78% increase in recommendation accuracy and 34.38% increase in recommendation diversity compared with the original RippleNet when using the sparse KG. When using the dense KG, both RippleNet and MKR perform worse but the utilization of context and interest can still improve the results.

Table 4. RippleNet and MKR performance comparison

Model	Context and interest utilization	Sparse KG		Dense KG	
		AUC	ACC	AUC	ACC
RippleNet	without context and interest	0.6128	0.6071	0.5055	0.5391
MKR		0.5590	0.5741	0.4616	0.4648
RippleNet	with context and interest	0.8235	0.6786	0.5545	0.6428
MKR		0.6524	0.6073	0.4706	0.4779

To show more clearly how the sparse and dense KGs perform in knowledge recommendation, we select different parts of training data and test the RippleNet-based CKGRN with context awareness and interest model for multiple times. The results are shown in Figure 5. The average AUC was 0.82 for the sparse KG and 0.55 for the dense KG. The average ACC was 0.72 for the sparse KG and 0.66 for the dense KG. It is evident that while the dense KG modeling fine-grained semantic relations between knowledge items can better support knowledge inference and retrieval, it is less effective in supporting knowledge recommendation compared with the sparse KG.

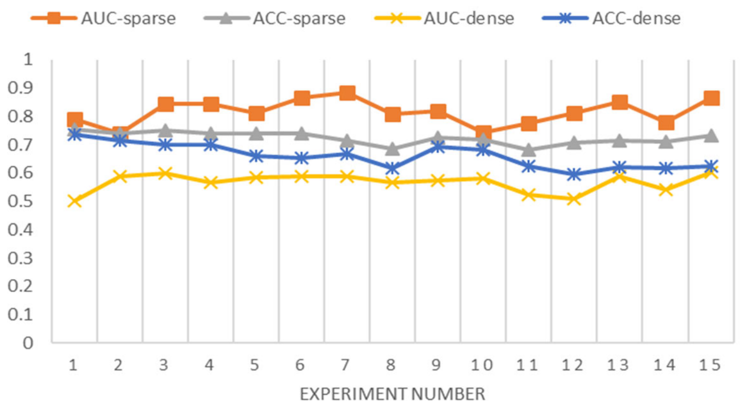

Figure 5. Performance comparison of the sparse and dense KGs

The defectiveness of dense KG in supporting shipbuilding knowledge recommendation can be attributed to two aspects. First, the embeddings of nodes and relations in a dense knowledge graph may not be sufficiently learnt due to insufficient training data and limitation on ripple set size at each hop. Second, through computing the degree centrality of nodes in the dense KG, we obtain the nodes with the maximum degree centrality as shown in Table 5. In the dense KG, connections between knowledge items primarily rely on semantic relations between nodes representing phrases in the knowledge items, and phrase nodes with higher degree centrality occupy crucial hub positions. However, from Table 5 we can see that the top-ranked hub nodes are often generic and not representative, meaning they do not truly represent a specific knowledge item. Therefore, when neural networks activate these nodes during iterations, these nodes, due to their key positions, gain higher recommendation weights. This can push out other nodes representing needed knowledge, leading to a decrease in the accuracy of knowledge recommendation.

Table 5. Max degree centrality of nodes in the dense KG

No.	Node name	Degree centrality
1	design	11
2	mm	7
3	require	7
4	not	7
5	modify	7
6	cannot	7
7	increase	6
8	consider	6
9	fit	5
10	large	5
11	cable	5
12	model	5
13	deploy	4
14	block	4
15	affect	4

4. CONCLUSION

Given the sparse interactions between users and knowledge items within a task-focused group characterized by dynamic tasks, it's essential for the recommendation process to leverage not just the direct user-item interactions but also the underlying relationships among the knowledge items themselves. KG-based recommendation models become potent solutions to this problem since a KG can describe the relations between knowledge items clearly. However, among the established models, few have discussed how to build a KG suitable for knowledge recommendation. In this paper, we show two ways of building KGs for a shipbuilding enterprise, and use two advanced recommendation algorithms, RippleNet and MKR, to do KG-based knowledge recommendation. Context awareness and a user interest model are also used to improve the performance of knowledge recommendation.

Through carrying out comparative experiments with differently structured KGs and recommendation algorithms, we identify a combination of the two aspects which can achieve better knowledge recommendation in the shipbuilding scenario. An important discovery is that a dense KG modeling fine-grained semantic relatedness between knowledge items is not so effective in knowledge recommendation compared with a relatively sparse KG constructed from the knowledge taxonomy manually compiled by experts in the shipbuilding enterprise. This may be caused by underfitting of KG embeddings given limited training samples combined with limited ripple set size, but the content of the dense KG definitely plays a role in determining the recommendation result.

The effectiveness of an automatically constructed KG in supporting knowledge recommendation is heavily influenced by the machine learning algorithms used during the KG construction. If the algorithms employed in extracting entities and relations are not accurate, the efficacy of the recommendation can be compromised. This issue underscores the importance of having domain experts review both the process and outcomes of automatically constructed KGs. Such oversight ensures that the entities and relationships extracted reflect the core content of the knowledge items and their contextual relevance within the application.

REFERENCES

Baltrunas, L. et al., 2012. Context relevance assessment and exploitation in mobile recommender systems. *Personal and Ubiquitous Computing*, Vol. 16, pp. 507-526.

Cai, X. et al., 2022. Explicable recommendation based on knowledge graph. *Expert Systems with Applications*, Vol. 200, 117035.

Chang, Y. et al., 2023. Meta-relation assisted knowledge-aware coupled graph neural network for recommendation. *Information Processing & Management*, Vol. 60, No. 3, 103353.

Gao, M. et al., 2023. Enhanced multi-task learning and knowledge graph-based recommender system. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 35, No. 10, pp. 10281-10294.

Guo, Q. et al., 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 34, No. 8, pp. 3549-3568.

Ji, Y. et al., 2023. A multitask context-aware approach for design lesson-learned knowledge recommendation in collaborative product design. *Journal of Intelligent Manufacturing*, Vol. 34, No. 4, pp. 1615-1637.

Kaur, G. et al., 2023. A deep learning knowledge graph neural network for recommender systems. *Machine Learning with Applications*, Vol. 14, 100507.

Kejriwal, M., 2022. Knowledge graphs: a practical review of the research landscape. *Information*, Vol. 13, No. 4, 161.

Li, Q. et al., 2022. Improving entity linking by introducing knowledge graph structure information. *Applied Sciences*, Vol. 12, No. 5, 2702.

Ma, T. et al., 2023. KR-GCN: Knowledge-aware reasoning with graph convolution network for explainable recommendation. *ACM Transactions on Information Systems*, Vol. 41, No. 1, pp. 1-27.

Song, B. et al., 2016. Automated experiential engineering knowledge acquisition through Q&A contextualization and transformation. *Advanced Engineering Informatics*, Vol. 30, No. 3, pp. 467-480.

Sun, K. et al., 2021. Context-aware seq2seq translation model for sequential recommendation. *Information Sciences*, Vol. 581, No. 9, pp. 60-72.

Vizcarra, J. et al., 2024. 'Representing the interaction between users and products via llm-assisted knowledge graph construction', in *2024 IEEE 18th International Conference on Semantic Computing (ICSC)*. Laguna Hills, CA, USA, pp. 231-232.

Wang, H. et al., 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. *Proceedings of the 27th ACM international conference on information and knowledge management*. Torino, Italy, pp. 417-426.

Wang, H. et al., 2019. 'Multi-task feature learning for knowledge graph enhanced recommendation', in *The world wide web conference*. San Francisco, USA, pp. 2000-2010.

Wang, L. K. et al., 2022. Context-aware GAN-based knowledge recommendation method in engineering field. *Computer Integrated Manufacturing Systems*, Vol. 28, No. 3, pp. 798-811.

Wang, X. et al., 2021. Learning intents behind interactions with knowledge graph for recommendation. *Proceedings of the web conference 2021*. Ljubljana, Slovenia, pp. 878-887.

Zammali, S. and Yahia, S. B., 2021. How to select and weight context dimensions conditions for context-aware recommendation?. *Expert Systems with Applications*, Vol. 182, 115176.

Zhao, N. et al., 2023. AGRE: A knowledge graph recommendation algorithm based on multiple paths embeddings RNN encoder. *Knowledge-Based Systems*, Vol. 259, 110078.