Information Sciences 613 (2022) 69–95

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Similarity attributed knowledge graph embedding enhancement for item recommendation

Nasrullah Khan [a,e], Zongmin Ma [a,b,*], Aman Ullah [c,f], Kemal Polat [d]

[a] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
[b] Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, 210000, China
[c] Yangtze Delta Region Institute, University of Electronic Science and Technology of China, Huzhou 313001, China
[d] Department of Electrical and Electronics Engineering, Bolu Abant Izzet Baysal University, Bolu 14280, Turkey
[e] Department of Computer Science, COMSATS University Islamabad, Vehari Campus, Vehari 61100, Pakistan
[f] School of Computer Science and Engineering, Central South University, Changsha 410083, China

ARTICLE INFO

Article history:
Received 19 September 2021
Received in revised form 28 August 2022
Accepted 31 August 2022
Available online 14 September 2022

Keywords:
Knowledge graph
KGE and KGEE
Recommender systems
Similarity
Interactions
Features

ABSTRACT

Knowledge Graph Embedding (KGE)-enhanced recommender systems are effective in providing accurate and personalized recommendations in diverse application scenarios. However, such techniques that exploit entire embedded Knowledge Graph (KG) without *data relevance* approval constraints fail to stop noise penetration into the data. Additionally, approaches that pay no heed to tackle *semantic relations* among entities remain unable to effectively capture semantical structure of Heterogeneous Information Graph (HIG). Therefore, in this paper, we propose **S**imilarity **A**ttributed **G**raph-embedding **E**nhancement (SAGE) approach to model similarity-aware semantic connections among entities according to their triplets' granularity. SAGE is a novel Knowledge Graph Embedding Enhancement (KGEE) method that constructs *Entity-relevance*-based Similarity-attributed Subgraph (ESS) to remove noise from the underlying data. It propagates interactions-enhanced knowledge over ESS to learn higher-order semantic connections among entities; and simultaneously utilizes feedbacks to enhance the interactions and regularize the model to highlight influential targets (nodes). Further, it samples influential targets in KG, independently move their preferences to the Local Central Nodes (LCN) of current influential areas, and streamline the collected information from all LCN to the main unit. Finally, a prediction module is used to determine generalized preferences for recommendation. We performed extensive experiments on benchmark datasets to evaluate the performance of SAGE where it outperformed the state-of-the-art methods with significant improvements in effectively providing the desired explainable recommendations.

1. Introduction

Current advancement of internet technology has not only enhanced e-commerce and its applications, but also inundated the online-world with similar options. Data-overload on the web makes the selection of suitable choices time consuming and sometimes even challenging. Recommender System (RS) is used to rationalize the data-overload and suggest suitable choices to the end-users to satisfy their needs and heeds. Recommendation techniques exploit current information (i.e., current

* Corresponding author at: College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China.
E-mail address: zongminma@nuaa.edu.cn (Z. Ma).

https://doi.org/10.1016/j.ins.2022.08.124
0020-0255/

interactional-networks and previous interaction-records of the users) to provide future recommendations. Specifically, Collaborative Filtering (CF)-based RS uses interactions history of the users to predict their preferences about unseen items of their interests [1]. Although these approaches are popular and widely used in recommendation, their performance is significantly affected by cold-start and data-sparsity problems [1]. Moreover, traditional CF-based methods fail to provide sufficient reasoning for explainable recommendations since they entertain explicit interactions among users and items only and overlook the rest of the information necessary for explainable inferences [2,3]. Therefore, various approaches have incorporated user-item side information into recommendations to overcome the breaches. For instance, [4] used plain-text-based user-item information, [5] extracted entity information from social networks, [6] exploited location-information, and [7,8] used contextual information. KG-based side information is deemed as the most feasible type of side information that exhibits facts and figures (i.e., data) through entities and relations among them. *The process of translating (converting) KG-based data into the low-dimensional vectors (*i.e., *embedding space) is known as KGE*, and *the degree of relatedness of the underlying datasets with the information of application scenarios is known as information relevance*. *KGE* has significantly contributed to recommendations by efficiently providing the heterogeneous information to RS [9,10], but the importance of *information relevance* is always overlooked that affected the computation and declined the performance.

KG-based recommendation relies on some well-known benchmark datasets[1] due to their easier availability, acquisition and utilization. These clearly labeled datasets time to time retrieve massive data from online knowledge repositories[2] via various data retrieval techniques[3] to keep their knowledge bases up to date. Consequently, irrelevant data is also retrieved with the main contents that expands the scope of underlying information to various topics besides the actual labels of the data – causes issues of noise emergence and data sparsity. For instance, in Fig. 1, if everything excluding this meta-path "*Ahmad → OOP in C++ → Programming → OOP with Java*" is irrelevant, the incoming and outgoing data in all of the meta-paths via the node "*Programming*" is noise to the context of *computer programming*. We try to explain this scenario with the help of some meta-paths that are irrelevant to the context. For example, "Winter Olympics → Reception → 160 Hours → Television → Programming → Ten days", "Doctors → Endurance → Programming → Cardiovascular Disease → Infancy", "Kingsley → Operations → Daytime → Programming → Air → Shows", "Flaw → Computer → Programming → Glitches → Database", "Instinctive → Mental → Programming → Directions → Work → Creatures". These meta-paths contain noise since incoming and outgoing information on "Programming" is not relevant to the context. Moreover, concerning personalized recommendations that are highly delicate to the data sparsity issues, fixed-scope datasets[4] are used to provide scenario-based instant recommendations in the current approaches. Thus, similarity-based data filtration is compulsory to enhance applicative significance of the underlying data.

Connections among nodes are channels to infer reasoning for recommendations and RS commonly exploits these channels to process KG-based side information [11]. Path and propagation-based methods are, therefore, well-known and frequently applied techniques in KG-based recommendations [3]. Although these approaches are decent in attaining reasonable performance [11,12], their excellence is hindered by several notable limitations they suffer from. Typically, path-based methods [13,14] model meta-paths and use their mutual similarities to determine more identical paths in the given data. This process is normally constrained to a limited path length (p) to avoid computational time complexity. The trade-off between p and complexity results in information loss. For instance, a length constraint (e.g., a meta-path of p relations is entertain-able only where p is a number) is imposed on sampling length of the meta-paths and the relations exceeding the given length are discarded. Consider a meta-path "*Ahmad → Asad → H/F Java → Publisher → H/F Java-Script*" from Fig. 1 where "*→ H/F Java-Script*" will be discarded if p is equal to 3. Moreover, due to the linearity concern of the meta-paths, these methods are unable to efficiently capture semantical and hierarchical structures of HIG. Similarly, propagation-based methods [12,15] propagate data over the whole graph and aggregate the required information from all entities in the graph. The point worth notifying is researchers overlook the presence of implicit or explicit interactions in the propagated data with respect to (wrt) entities of graph – a data relevance concern. For instance, let "*Ahmad*" interacts with "*OOP with Java*" via the meta-path "*Ahmad → OOP in C++ → Programming → OOP with Java*" and the rest of the possible meta-paths start from "*Ahmad*" are un-interacted. It implies that the only interacted item (i.e., "*OOP with Java*") describes the interest of "*Ahmad*" via the concerned meta-path. Although these methods acquired great attention, inspiration and popularity for their attained performance, effective incorporation of HIG into RS is challenging from the perspectives of information relevance, structural and semantics concerns.

To effectively address the highlighted challenges, we propose SAGE – a novel similarity attributed KGEE approach for item recommendation – in this paper. Formally, SAGE is comprised of following four functional modules: (a) *Similarity-guided Subgraph Construction* (SSC) is designed to capture semantically relevant higher-order connections among entities in the data and construct ESS. SSC is a similarity-based neighborhood computation technique that folds higher order relevant neighbors in its influential graph. (b) *Interaction-enhanced Knowledge Network* (IKN) is designed to propagate [12] implicit interaction-based entity-information over ESS. (c) *Feedback-attributed Interaction Enhancement* (FIE) technique is introduced to simultaneously integrate explicit feedback features into the implicit interactions and enhance system's regularization [16] capability

[1] e.g., Book-Crossing, Amazon-Book, Amazon-Products, Movie-Lens, Last-FM, Bing-News, Yelp, etc.
[2] e.g., DBpedia, CN-DBpedia, YAGO, Wikidata, Freebase, NELL, Google's KG, Microsoft Satori, etc.
[3] Web-crawling, Mapping, Extracting, Curation, etc.
[4] That are usually specific to their titles, such as, Book-Crossing, Last-FM and Yelp, in our case.

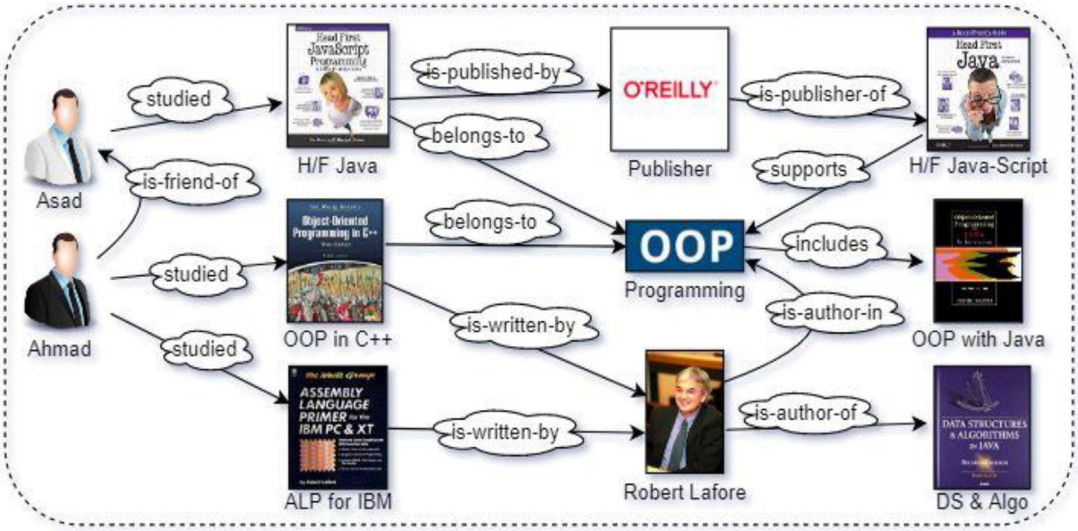


Fig. 1. Working paradigm of Book Recommendation process based on Book-Crossing Dataset where Users (i.e., *"Ahmad"* and *"Asad"*) interact with Entities (e.g., *"ALP for IBM"*, *"OOP in C++"*, *"Asad"* and *"H/F Java"*, etc.) and Books (i.e., *"H/F Java-Script"*, *"OOP with Java"* and *"DS & Algo"*) are recommended to them, as shown in Fig. 10.

to highlight the influential targets in IKN through Graph Neural Network (GNN). FIE regularizes the required information in higher-order neighborhood via a long-range of indirect dependencies. (d) *Sampling, Retrieval and Formalization* (SRF) suite is configured to sample the highlighted influential targets, independently collect and move the required preferences from the influential targets to the LCN of current influential communities, and streamline all LCN to the main unit through gated formalization technique. To the best of our knowledge, SAGE is the first ever contribution to KGE against noise penetration and KGEE-based item recommendation.

The key offerings of this work are listed as follows:

- We propose a novel *Node Similarity-guided Path* (NSP) *modeling* technique to construct ESS to optimally keep the underlying data noise free.
- We design IKN to propagate implicit interaction-based entity-information over ESS. We propose FIE technique to incorporate explicit feedback features into the implicit interactions to concurrently enhance IKN and system's regularization capability to highlight the influential targets in IKN.
- We Introduce SRF suite to sample the highlighted influential targets, collect and move their preferences to LCN, and streamline all LCN to the main unit.
- We performed extensive experiments on real-world benchmark datasets to evaluate the effectiveness of SAGE. The results' analysis demonstrates that the proposed approach has outperformed the baseline methods with remarkable improvements by smoothly meeting the aforementioned challenges.

The rest of the paper's organization is as follows: Section 2 describes the related work, Section 3 presents the preliminary concepts, Section 4 demonstrates the proposed methodology, Section 5 summarizes the experiments and the empirical work, and finally, section 6 concludes the paper.

2. Related work

In this section, we categorically overview the literature that is closely related to the proposed approach.

2.1. Path-based recommendation methods

Path-based methods exploit the meta-paths of previous interactions to guess similar meta-paths for future recommendations based on probabilistic reasoning [17–20]. For instance, Personalized Entity Recommendation (PER) [9] modeled inter-entity relations through meta-path-based features extraction. Based on mutual similarity of the meta-paths, PER learned latent features of user-item interactions to calculate user-preferences. SemRec[5] [13] employed the measures of similarity among meta-paths connecting users and items to provide personalized recommendations. It also utilized the concept of

[5] Semantic Path-based Personalized Recommendation.

weighted meta-paths to predict user-ratings about the targets interacted in the past. Interactive-Rules-based Recommendation (IR-Rec) [2] proposed an embedded path-aware rules-guided approach for KG-based explainable recommendations. It converted the extracted paths to the predefined rules to describe the potential motivations of users. Moreover, adversarial-regularization technique is proposed to decrease the level of noise in the data and jointly used attention mechanism with bidirectional long short-term network through meta-path-based inferential reasoning for recommendation [18]. Recurrent KGE (RKGE) [31] and Knowledge-aware Path Recurrent Network (KPRN) [11] mined meta-paths with specific lengths and weights automatically, and exploited them via knowledge aware path recurrent neural network. Furthermore, [19] utilized meta-path enhanced approach to capture higher order relations in the data for sequential recommendation. MCRec[6] [20] introduced rich meta-path concept to acquire the context representations of users and items via aggregation of neural-interaction model with co-attention mechanism for top-K recommendation. Factorization Machine with Group-lasso (FMG) [23] exploited the measures of similarity among paths/meta-paths connecting users and items. Kang et al. [22] exploited the interaction track of user-log as global information through the combination of self-attentive technique with topic modeling for sequential recommendation. It is hereby worth clarifying that none of the above discussed methods have applied any information relevance verification constraint.

2.2. KGE-based recommendation methods

In KGE, the data (i.e., information instances and their internal dependencies) is converted to the vector space to obtain feature vectors that are used as side information in recommendation [24,25]. Collaborative Knowledge base Embedding (CKE) [26] jointly learned items' semantic embedding from the designed knowledge-base and latent representations via CF-technique in a combined Bayesian framework for preference estimation. CKE applied TransR for KG translation. HERec[7] [27] and HIN2Vec[8] [28] used KG-based random walk sampling technique to obtain and maintain latent representations of node-sequences for network embedding. Relational Collaborative Filtering (RCF) [29] jointly used DistMult to learn inter-item relations and attention mechanism to access user preferences for recommendations, and KTUP[9] [30] used TransH and jointly learned recommendation and KG-completion modules. Deep-latent approaches [10,21] are proposed to fix data sparsity problem via data segmentation and improved latent-factor based on the integration of topological similarity-measures into the model. KG-enhanced Sequential Recommender (KSR) [14] aggregated a memory network with RNN and enhanced semantical representations of KG-based information instances for sequential recommendation. KSR used TransE for KG conversion. Deep Knowledge-aware Network (DKN) [32] integrated data-semantics into the entity-representations via knowledge-aware CNN for news recommendation. DKN used TransD to accomplish feature learning-based KG embedding. LR-GCCF[10] [33] tried to improve the performance of CF-based methods with Graph Convolutional Network (GCN) through (i) eliminating non-linearities in the data, and (ii) proposing residual graph structure to model user-item interactions intended to remove the limitations of over smoothing in convolutions' aggregation during the sparse interactions. Similarly, [1] combined CF-algorithms with attention mechanism and GCN to design a KG enhanced neural CF-framework for modeling user-item interactions. The proposed approach is used to address data sparsity and enhance recommendation performance.

2.3. Propagation-based recommendation methods

Numerous recent KG-based recommendation approaches performed information propagation over the data and effectively obtained the required preferences. For instance, RippleNet [12] propagated previous user-item interactions over KG to collect user-preferences for future recommendations. Deep Knowledge Enhanced Network (DKEN) [16] is a cross-information-enhanced multilayered deep KG-based recommendation approach designed to integrate KGE into the RS model. DKEN also performed interactions' propagation over the network to collect the preferences. Moreover, [34,35] performed data propagation to learn entity embeddings wrt neighbor information in the KG without and with label smoothness (LS) regularization respectively. LightGCN [36] claimed to improve the designing structure of GCN for "neighborhood aggregation" wrt the CF-based methods for recommendation. It performed linear propagation on the graph to learn entity embeddings independently and collected a "weighted sum" of the embeddings from all processional layers as the ultimate embedding. KG Attention Network (KGAT) [15] recursively executed the propagation to enhance the embeddings and Attentive KGE (AKGE) [37] accomplished relation aware propagation over KG to improve the performance. Similarly, Neighborhood Aggregation CF (NACF) [38] incorporated the interests of users wrt their neighborhood information into the concerned data and propagated it on the network to learn entity embeddings. Although the discussed methods are good in providing the required recommendations, they totally overlooked the importance of information relevance concerning the adopted experimental data.

[6] *meta*-path based Context for Recommendation.
[7] Heterogeneous information network Embedding for Recommendation.
[8] Conversion of Heterogeneous Information Network (HIN) to Vector form.
[9] KG-enhanced Translation-based User Preference approach.
[10] Linear Residual Graph Convolutional CF.

3. Preliminaries

In this section, we describe preliminary notations, task formulation and a few important definitions.

3.1. Notations

We represent the sets of m-users and n-items via $\mathscr{U} = \{u_1, u_2, \cdots, u_m\}$ and $\mathscr{V} = \{v_1, v_2, \cdots, v_n\}$ respectively, the user-item interaction matrix via $\mathscr{M}|\mathscr{M} \in \mathbb{R}^{m\times n}$, and the interactions stored in $\mathscr{M}$ via $\mathscr{I}$ as $\mathscr{I}_{u,v}$ is 1 if there exist an interaction between u and v and 0 otherwise. Moreover, $\mathscr{I}_{\imath}$ and $\mathscr{I}_{\not p}$ are recent and previous interactions respectively, and we symbolize the set of node-neighbors via $\mathscr{N}(\Gamma)$ where Γ shows a node.

3.2. Task formulation

Top-K recommendation about the items of users' interests are required from the SAGE. Intuitively, the model considers inter-entity interactions as implicit interactions and ratings or feedbacks as explicit interactions; while $\mathscr{G}_s$ and $\mathscr{I}_\ell$ as the sources of concerned entities and their mutual interactions respectively. Therefore, SAGE uses its prediction function $\widehat{\delta}_{(u,v)} = \varphi\Big[(n_u, n_v) \otimes \Big(\mathscr{G}_s, \mathscr{M}_{uv}, \Theta, \bar{\boldsymbol{h}}_{uv}(n_u, n_v)\Big)\Big]$ to predict the extent of possibility that "*a user u_i will choose an unseen item v_j in his/her future selection*" based on the provided information, where φ is the prediction mechanism, Θ is the set of model parameters, $\bar{\boldsymbol{h}}$ shows the hidden units of the corresponding entities, $\otimes$ represents the binding parameters of the model, $n_u \& n_v$ are user's and item's nodes respectively and $\mathscr{G}_s$ is the subgraph. Finally, a score list generated by the prediction function is displayed in descending order to pronounce top-K recommendation about the items.

3.3. Definitions

In this section, we formally define a few important and technical terminologies to be used in this paper.

3.3.1. Knowledge graph and meta-Path

Knowledge graph is a collection of four elements that is defined as $G = (E, \mathscr{P}, \Phi, \Psi)$, where E represents the set of entities, $\mathscr{P}$ shows the set of paths among entities, Φ denotes entity-type mapping function f_e as $\Phi_{f_e} : E \rightarrow$ E, and Ψ describes path-type mapping function f_p as $\Psi_{f_p} : \mathscr{P} \rightarrow \mathscr{R}$. Each entity $e \subseteq E$ is mapped to a specific entity-type in E, i.e., $\Phi_{f_e}(e) \in \mathscr{E}$, and each path $p \subseteq \mathscr{R}$ is mapped to a concerned path-type in $\mathscr{R}$, i.e., $\Psi_{f_p}(p) \in \mathscr{R}$. If $|\mathscr{E}| > 1$ or $|\mathscr{R}| > 1$,G is heterogeneous otherwise it is homogenous.

Meta-path $\wp$ is a sequenced collection of finite number of paths p in G and p describes a single relation r between two consecutive nodes. For instance, a meta-path between nodes n_a and n_d is written as $\wp(n_a, n_d) = p_1^\circ p_2^\circ p_3$ that is equal to $\wp = n_a \xrightarrow{p_1} n_b, n_b \xrightarrow{p_2} n_c, n_c \xrightarrow{p_3} n_d$ implies that $\wp = n_a \xrightarrow{p_1} n_b \xrightarrow{p_2} n_c \xrightarrow{p_3} n_d$, where $\circ$ is a composition operator.

3.3.2. Interaction matrix

Interaction matrix $\mathscr{M}$ is comprised of implicit user-item interactions in the form of 0 s and 1 s, where 0 means there is no observed interaction between user u and item v and 1 represents an observed interaction. Objective expression of the interaction matrix is written as:

$$m_{uv} = \begin{cases} 1, & \textit{if interaction exists} \\ 0, & \textit{otherwise} \end{cases} \tag{1}$$

Where $m_{uv} \in \mathscr{M}$, $u \in \mathscr{U}$ and $v \in \mathscr{V}$.

3.3.3. Interaction log

Interaction log $\mathscr{I}_\ell$ is a timespan-based collected series of user-item interactions that are stored in different track of $\mathscr{I}_\ell$. Formally, $\mathscr{I}_\ell$ contains the sets of items $\mathscr{V} = \{v_1, v_2, \cdots, v_n\}$, interactions $\mathscr{I} = \{\imath_1, \imath_2, \cdots, \imath_t\}$, sessions $\mathscr{S} = \{\jmath_1, \jmath_2, \cdots, \jmath_t\}$ and timespans $T = \{t_1, t_2, \cdots, t_t\}$, where $\mathscr{I}$ contains the types of interactions such as viewed, rated, purchased, etc., $\mathscr{S}$ shows the interaction-sessions, and T represents the timespans of interactions. During an arbitrary session $\jmath_j = (v_j, \imath_j, t_j) \mid \jmath_j \in \mathscr{V} \times \mathscr{I} \times T$ or $\jmath_j = (v_j, \imath_j)|\jmath_j \in \mathscr{V} \times \mathscr{I}$ respectively, T is either implicitly extracted from $\mathscr{S}$ or explicitly collected from $\mathscr{I}$.

3.3.4. Tendency-based similarity

Tendency-based Next-node Prediction (TNP) determines the possibility of whether a current-node n_c could be connected to its 1-hop neighbor (i.e., target-node n_ℓ) or not via the following objective function:

$$TNP(n_c, n_\ell) = \frac{f(tendency)_{n_c \sim n_\ell}}{\sum_{\forall \Gamma | \Gamma \in \mathcal{N}(n_c)} f(tendency)_{n_c \Gamma}} \tag{2}$$

Where f shows the factor of tendency-based relevancy of n_c with n_ℓ defined as:

$$f(tendency)_{n_c \sim n_\ell} = \alpha \cdot \mathcal{I}_{\imath}(tend) + (1 - \alpha).\mathcal{I}_{\hbar}(tend) \tag{3}$$

Where $\alpha | 0 < \alpha < 1$ shows the tendency co-efficient and *tend* represents the tendency parameter.

The selection of n_ℓ is highly dependent on the inputs of $\mathcal{I}_{\imath}$ and $\mathcal{I}_{\hbar}$ since $\mathcal{I}_{\imath}$ and $\mathcal{I}_{\hbar}$ carry the interactions of previous one and six months respectively.

3.3.5. Appearance-based similarity

Among the nodes, local similarity Sim_{loc} depends on two aspects of n_c, (i) similarity between n_c and p going out of n_c, (ii) similarity between n_c and n_ℓ.

For (i), we used cosine-similarity to find the similarity between n_c and p going from n_c to n_ℓ as:

$$C(n_c, p) = \frac{\sum_{i=1}^{n} n_{ci} p_i}{\sqrt{\sum_{i=1}^{n} n_{c_i}^2} \sqrt{\sum_{i=1}^{n} p_i^2}} \tag{4}$$

For (ii), we unified Adamic-Adar and Jaccard (A&J) similarities to find the similarity between n_c and n_ℓ as:

$$A\&J(n_c, n_\ell) = 0.5 * \left(\sum_{y \in (\mathcal{N}(n_c) \cap \mathcal{N}(n_\ell))} \frac{1}{\log |\mathcal{N}(y)|} + \frac{\mathcal{N}(n_c) \cap \mathcal{N}(n_\ell) + 1}{\mathcal{N}(n_c) \cup \mathcal{N}(n_\ell) + 1} \right) \tag{5}$$

Finally, we added Eq. (4) and (5) to get the total appearance-based similarity as:

$$Sim_{loc}(n_c, n_\ell) = C(n_c, p) + A\&J(n_c, n_\ell) \tag{6}$$

3.3.6. Influential graph

Influential graph $\mathcal{G}_s | \mathcal{G}_s \in G$ is a network of similar nodes interconnected based on the relevance factor defined in the following expression:

$$f(x)_{\forall \Gamma | \Gamma \in \mathcal{N}(n_c)} = \begin{cases} 1, if \mathcal{G}_s(n_c) \geq \vartheta \\ 0, if \mathcal{G}_s(n_c) < \vartheta \end{cases} \tag{7}$$

Where 1 shows the presence of an influential connection between two neighbors, 0 represents no connection, ϑ is threshold-value (i.e., $\vartheta = 30\%$ in this case), $f(x) \in \mathcal{G}_s$ and $\mathcal{G}_s(n_c)$ is defined as:

$$\mathcal{G}_s(n_k) = \bigwedge_{i=1}^{s} \{ n_{k-i} | sim(n_k, n_{k-i}) > sim(n_k, n_{k-J}) \} n_{k-i}, n_{k-J} \tag{8}$$

Where $n_k | n_c = n_k$ is a temporary variable, $\wedge$ is the sequential concatenation operator, n_{k-i} shows the set of nodes connected to n_k, and n_{k-J} represents the set of nodes not-connected to n_k.

In the connected network (i.e., n_{k-i}), n_c is explicitly connected to all of its neighbors, i.e., $\mathcal{N}(n_c)$.

3.3.7. Swish

Swish, derived from [39], is a new self-gated non-linear activation function that performs well with the deep learning models. Formally, it is a dot-product of *the given information* and *the sigmoid of the given information* written as:

$$f(x) = x \cdot \sigma(x) \tag{9}$$

where σ shows the sigmoid function that can be defined as $\sigma(x) = (1 + exp(-x))^{-1}$.

3.3.8. Top-K recommendation

Top-$K | K = \widehat{\delta}_{(u,v)}$ is a list of recommendation results sorted in descending-order carries items that are to be selected by the users in their future selections, where $u \in \mathcal{U}$ and $v \in \mathcal{V}$ and $\widehat{\delta}$ shows u-v preference scores.

4. The proposed methodology

In this section, we describe the implementation details of the proposed approach.

4.1. Overview of the framework

"*Similarity-guided Subgraph Construction* (SSC)", "*Interaction-enhanced Knowledge Network (IKN)*", "*Feedback-attributed Interaction Enhancement (FIE)*" and "*Sampling, Retrieval and Formalization (SRF)*" are the main contributing modules of the SAGE framework as summarized in Fig. 2. SAGE receives the input data on its "*Data Reception*" port and transforms it to the vector (i.e., Embedding) space. The input data contains benchmark-datasets and user-log (i.e., previous record of the user-item interactions) as the underlying data to be used in experiments. User-item implicit interactions and explicit feedbacks are retrieved from benchmark-datasets and user-log respectively. After converting the data to the vector form, it is provided to the SSC module. The main purpose of this module is to stop noise penetration into the underlying data. Typically, information relevance check, threshold validation, and ESS construction are the basic responsibilities of SSC module. ESS is enriched by mapping its information instances to the relevant data-items in the external knowledge repositories by "*Data Enhancement*" and forwarded it to the IKN module.

IKN propagates interaction-enhanced user-item information over the graph to get user-preferences about the items of their potential interests. How the relevant information is extracted from the initial raw-data and user-log is discussed in Section 3.3.4 – Tendency-based Similarity. Moreover, in FIE, explicit interactions are separately extracted from ESS and used to concurrently enhance IKN and system's regularization capability to highlight the influential nodes in IKN. In terms of convenience, the influential nodes contribute more than the normal nodes during the preference collection since GNN provides a biased indicator to highlight these nodes.

The propagated information is normalized, collected, managed and provided to the "*Predictor*" for evaluation by the SRF module. Propagation-based recommendation methods used various aggregation techniques (e.g., neighbor, concatenation, sum, sum-product aggregators, etc.) to aggregate information. However, in the proposed approach, we introduce a new technique for preference aggregation, i.e., we apply inverse square law to sample the influential nodes in IKN. Further, the preferential information is independently moved from the sampled nodes to LCN of the current communities and applied gated-formalization technique to streamline all LCN to the main unit for independent information transaction. Finally, the formalized information is received and processed by the "*Predictor*" to generate recommendations.

4.2. Similarity-guided subgraph construction (SSC)

We pre-trained the data and projected the nodes and paths as embeddings to the Euclidian space based on paths as the semantic connections among nodes. For instance, (n_c, p, n_ℓ) is embedded as $n_\ell^p \cong n_c^p + n_p$ where n_c^p, n_ℓ^p are the embedding representations of n_c, n_ℓ in p's relation-space, and $n_c, n_\ell \in \mathbb{R}^d$ and $n_p \in \mathbb{R}^k$ are the embeddings of nodes and paths in the Euclidian distance respectively. Further, triplets from d-dimensional entity-space are transformed to k-dimensional relation-space via the following function:

$$\mathscr{g}(n_c, p, n_\ell) = \|\mathscr{T}_p(n_c - n_\ell) + n_p\|_2^2 \tag{10}$$

where $\mathscr{g}$ shows the grievance factor of the triplets, i.e., higher is the $\mathscr{g}$ greater is the invalidity of the triplets and vice versa, $\mathscr{T}_p \in \mathbb{R}^{d\times k}$ is the transformation matrix wrt to path p, and $\|\cdot\|_2$ is the Frobenius or Euclidian norm.

Though random walk-based path or meta-path sampling techniques are widely used to retrieve entity information from the data [14,17,28], the selection of feasible meta-paths requires a considerable assistance from the domain experts. Moreover, a trade-off between the length of meta-path and the amount of noise diffusion ever degrade the performance. To address these limitations, we propose *Node's Similarity-guided Path (NSP) modeling* technique, defined in Sections 3.3.4, 3.3.5 and 3.3.6, to automatically crawl the prominent paths in KG, as shown in Fig. 3. Technically, while having the walker on a temporary central node $\bar{n}_C$, we considered $\bar{n}_C$ as the current node n_c. NSP evaluates 1-hop neighbors of n_c to find a target node n_ℓ for the next step of the walker. The mechanism depends on higher is the similarity between n_c and the candidate node based on Eq. (11), more probable is its selection as n_ℓ. After the selection of the candidate node as n_ℓ, it is included to the influential graph, the path between n_c and n_ℓ is highlighted and n_ℓ is again considered as n_c to select the next node. The process is repeated up to hop-5, the highlighted meta-path (let say $\wp_1$) is termed as a prominent meta-path and identified via a unique tracking ID. The control is returned to the hop-1 community (i.e., $\bar{n}_C$) and selected the second most relevant neighbor of $\bar{n}_C$ as n_ℓ in the $\mathscr{N}\left(\bar{n}_C\right)$. The process is continued until the data is not converged. Consequently, ESS is achieved by discarding the un-highlighted nodes and paths from the KG, as shown in Fig. 3 and Algorithm 1. In the influential graph, only the colored nodes that are interlinked via the solid lines are kept and the uncolored nodes that are interconnected via the dotted lines are discarded. The solid colored lines are used to demonstrate a few examples of the selected meta-paths.

Mathematically, by adding Eq. (2) and Eq. (6), we got the objective function to calculate inter-node similarity as:

$$Sim_{node}(n_c, n_\ell) = TNP(n_c, n_\ell) + Sim_{loc}(n_c, n_\ell) \tag{11}$$

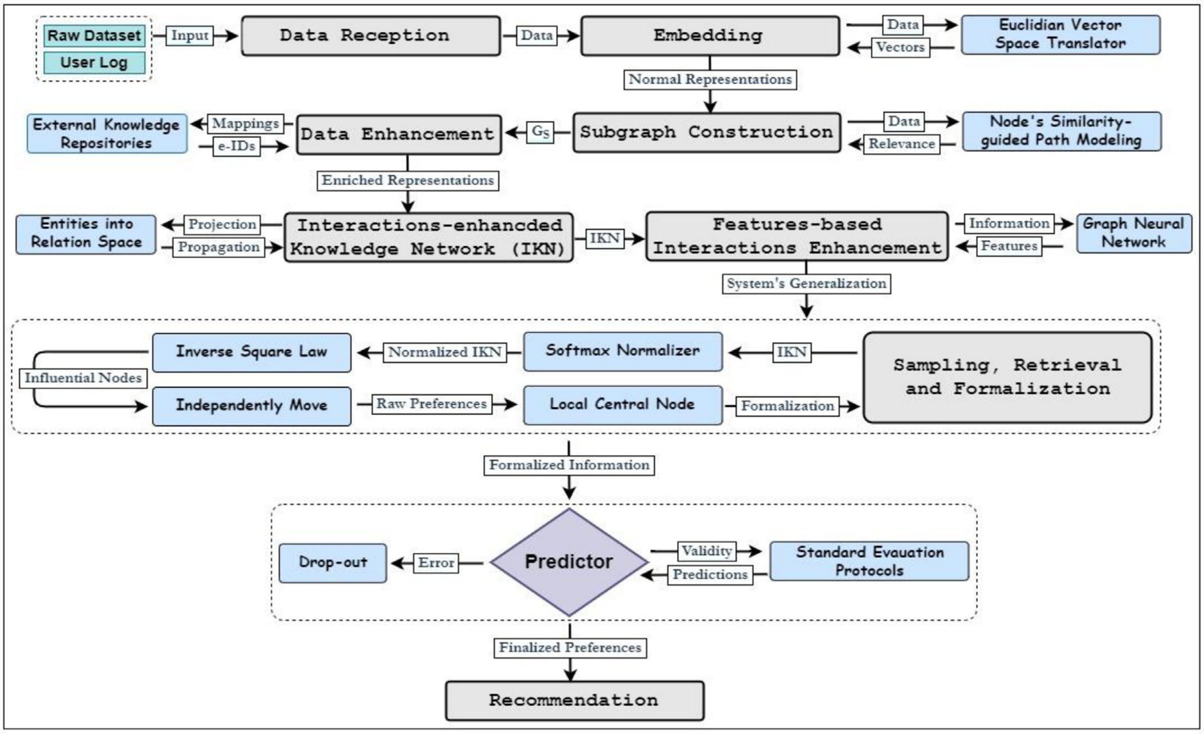


Fig. 2. Data and Control Flow Diagram of the SAGE's Framework.

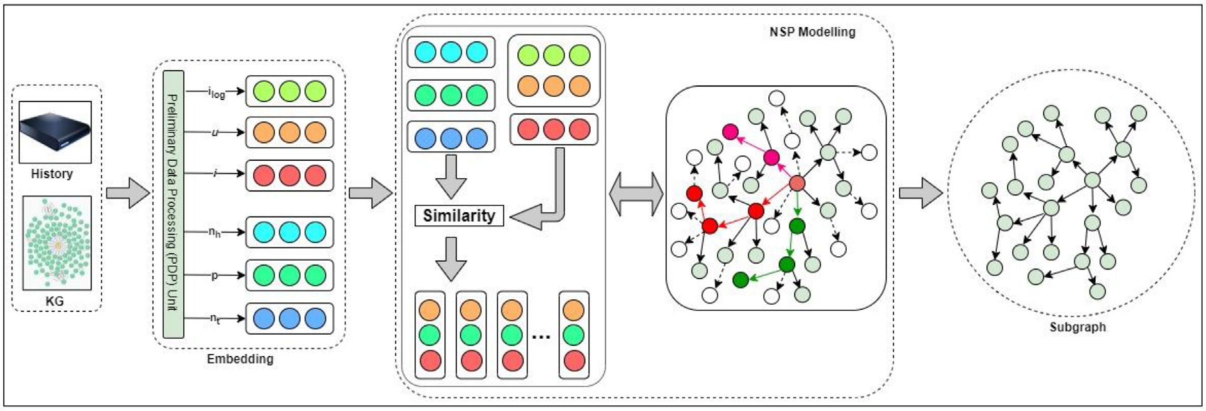


Fig. 3. Graphical presentation of NSP modeling-based meta-path selection and subgraph construction approach. In the box one step before the last circle representing the "Subgraph", the solid lines connecting the colored nodes are the selected meta-paths and the dotted lines connecting the uncolored nodes are the discarded meta-paths wrt the central nodes. The last circle shows a possible depiction of the ESS.

4.3. Interaction-enhanced Knowledge Network (IKN)

Paths carry information among nodes and depict rich semantics in the influential graph, but current methods propagate information in the network based on the entity embeddings only and ignore the importance of feasible path selection. Therefore, SAGE's *Interactions-guided Feasible Path* (IFP) selection and utilization is a bonus to the recommendation's performance, as demonstrated via Algorithm 2. Typically, we projected the entities n_i, their types n_i^τ and their interactions $\mathscr{I}_i$ to the relation spaces p_i and p_i^τ respectively to enhance the underlying knowledge of $\mathscr{G}_s$ through $\mathscr{I}_i$ wrt p as $p_{(a_i,b_i)\vdash \mathscr{I}_i}|p_{(a_i,b_i)} \in \mathbb{R}^{k^{v\tau}}$, where $n_i, \mathscr{I}_\ell \in \mathbb{R}^d$, $n_i^\tau \in \mathbb{R}^{d^\tau}$, $p_i \in \mathbb{R}^k$, $p_i^\tau \in \mathbb{R}^{k^\tau}$, and d, d^τ, k, k^τ are the respective sizes of the vector space, whereas $p_{(a_i,b_i)\vdash \mathscr{I}_i}|p_{(a_i,b_i)} \in \mathbb{R}^{k^{v\tau}}$ is a path p between nodes a_i and b_i enhanced by $\mathscr{I}_i$, $\vdash$ shows the affiliation of interactions with the corresponding triplet, $\mathbb{R}^{k^{v\tau}}$

is the embedding of $p_{(a_i,b_i)}$ and $k^{v\tau}$ is its embedding size. We kept the information instances interlinked with their concerned types in the relation space to maintain the enhancement of embeddings wrt their entity-types, relations and interactions as:

$$\widetilde{n}_i = \sigma\left(\mathscr{T}_n\left(n_i \parallel n_i^\tau\right) + \mathscr{T}_p\left(p_i \parallel p_i^\tau\right) \cdot \mathscr{I}_i + 2b\right) \tag{12}$$

where $\widetilde{n}_i$ is the embedding enhancement of n_i via n_i^τ, p_i, p_i^τ and $\mathscr{I}_i$, σ is the sigmoid function, $\mathscr{T}$ is a transformation matrix, b is a bias variable, and $\parallel$ is the integration operator.

The hidden state of $\widetilde{n}_i$ (i.e., h_i^t) is used to initiate the propagation from step $t = 0$ to *max* (i.e., till the convergence of data) wrt to the IFP. Therefore, we integrated the enriched path between the given nodes (i.e., enriched p between n_c and n_ℓ as $p_{(n_c,\ n_\ell) \vdash \mathscr{I}_\ell}$) into the concerned hidden-state as:

$$\bar{h}_\ell^t = \sigma\left(\mathscr{T}\left(\mathrm{h}_\ell^\mathrm{t} \parallel \mathrm{p}_{(\mathrm{n_c},\ \mathrm{n}_\ell)} \vdash\ \mathscr{I}_\ell\right)\right) \tag{13}$$

where $\bar{h}_\ell^t$ and h_ℓ^t show the enriched-hidden state and hidden state of the n_ℓ respectively, $p_{(n_c,n_\ell)} \in \mathbb{R}^{k^{v\tau}}$ and $n_\ell \in \mathscr{N}(n_c)$.

The indices of $\bar{h}_\ell^t$ are responsible to maintain the flow of information among nodes through the concerned paths to enrich the tail nodes. Thus, $\bar{h}_{\ell_i}^t$ keeps the track of information transferred between n_c and n_{ℓ_i} through the given path p_i as $p_{i\left(n_c,n_{\ell_i}\right)\mathscr{I}_{\ell_i}}$ to enrich n_{ℓ_i}, where $i = 1$ to 5. We demonstrate the information propagation among nodes and nodes' enhancement through a tiny example based on Fig. 5. Consider the meta-paths $\wp_\beta : n_5 \xrightarrow[\mathscr{I}_5]{p_5} n_8 \xrightarrow[\mathscr{I}_8]{p_8} n_{11}$ and $\wp_\gamma : n_4 \xrightarrow[\mathscr{I}_5]{p_5} n_8 \xrightarrow[\mathscr{I}_9]{p_9} n_7$, where node n_8 acts as a bridge in the meta-paths $\wp_\beta$ and $\wp_\gamma$. The successor nodes get information from predecessors and enrich their local representations, and likewise, the information and preferences are also propagated to the tail nodes. Moreover, n_8 enriches itself differently on different meta-paths, i.e., $n_5 \xrightarrow[\mathscr{I}_5]{p_5} n_8$ and $n_4 \xrightarrow[\mathscr{I}_5]{p_5} n_8$, and do not intermix the information received from different sources. Therefore, it provides different information to n_{11} and n_7 propagated via p_8 and p_9 respectively. Nearer the nodes in the meta-paths, stronger the information they receive and vice versa.

We configured IFP-based attention mechanism with Swish – a nonlinear activation function defined in Section 3.3.7 – to propagate information via the concerned paths as:

$$\pi(n_c, p, n_\ell) \vdash \mathscr{I}_\ell = f\left(y^\top \cdot x \cdot \sigma(x)\right) \tag{14}$$

where π is the softmax function, $f(y) = \mathscr{T}_p n_\ell \cdot \mathscr{I}_\ell$, $f(x) = \mathscr{T}_p n_c + \left(n_p \cdot \mathscr{I}_\ell\right)$, and $f(\cdot)$ is a function used to make the attention results reliant between the concerned nodes wrt to the embedded relation space.

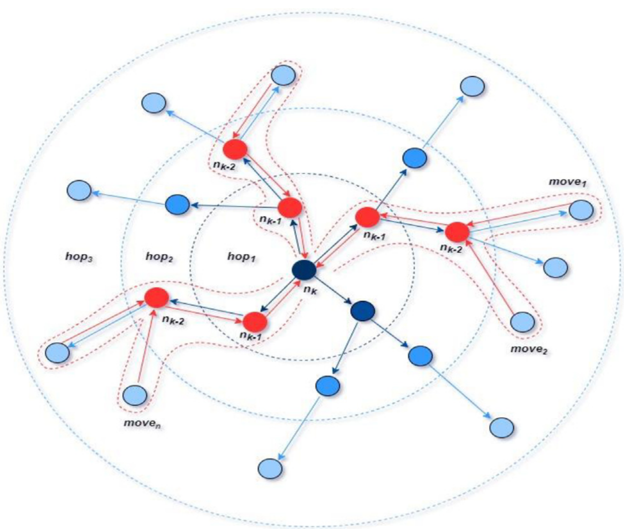


Fig. 4. Pictorial representation of the SRF Module. In this figure, n_k shows the main central node and n_{k-1}, n_{k-2}, etc. represent the LCN of the current influential communities. The ordinary nodes receive information from their 1-hop neighbors as well as move their current information to the LCN. Finally, the information from all LCN is streamlined to the main unit. The intensity of information moving inward is higher than the intensity of information moving outward like ripples or capillary waves on the surface of water when a stone is dropped in a lake. The outward fading color of the nodes describes the decreasing intensity of the information moving towards the higher-hops.

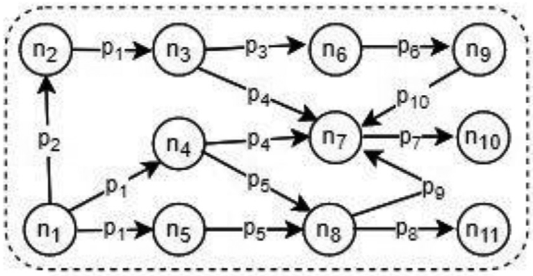


Fig. 5. Raw sketch of a few meta-paths from the Subgraph.

Algorithm 1. NSP Modeling-based Subgraph Construction

Inputs:
nodes, paths, $h = 0$, $H = 5$: Hop index and Hop length respectively, G: KG, $\mathscr{I}_\ell$: Interaction log, $\mathscr{M}$: Interaction Matrix, $\mathscr{N}$: Neighbors, d: embeddings.
Output: Similarity Attributed Subgraph.

1 **foreach** (n_c, n_ℓ) pair **do**
2 Reset h
3 **while** $h! \leftarrow H$ **do**
4 $n_\ell \leftarrow$ nearest neighbor of n_c
5 **if**$(\mathrm{cmp}(n_c, n_\ell) > \vartheta)$ according to $f(x)_{\forall\Gamma|\Gamma\in\mathscr{N}(n_c)} = \begin{cases} 1, if \mathscr{G}_s(n_c) \geq \vartheta \\ 0, if \mathscr{G}_s(n_c) < \vartheta \end{cases}$
6 **call** draw
7 **else**
8 $n_\ell \leftarrow$ next neighbor of n_c & **goto** line. 5
9 **end if**
10 **if** $|\text{no. of neighbors of } n_\ell)| > 0$
11 $n_c \leftarrow n_\ell$ & **inc** h
12 **else**
13 $n_\ell \leftarrow$ next neighbor of n_c & **goto** line. 5
14 **end if**
15 **end while**
16 identify the selected $\wp$ as $\wp_j$ & store $\wp_j$ in $\wp$-array
17 **end foreach**
18 **concat** $\wp$ from $\wp$-array wrt $\wp_j$
19 **proc** draw
20 n_ℓ : if not highlighted, highlight;
21 n_ℓ : if not added to $\mathscr{G}_s$, add;
22 **concat** (n_c, n_ℓ)
23 **end proc**

We need to linearize the influential graph based on the concerned meta-paths wrt n_c to hierarchically frame k-order path-connections in $\mathscr{G}_s$ as discussed in Section 4.2. Thus, the general objective function of path-connections' linearization is defined as:

$$\bar{n}_{\mathscr{N}(n_c)} = \bigwedge_{n_c,\ p,\ n_\ell \in \mathscr{N}(n_c)} \pi(n_c,\ p,\ n_\ell) \vdash \mathscr{I}_\ell \cdot \bar{n}_\ell \tag{15}$$

where $\bar{n}$, i.e., $\bar{n}_C$, shows a central node in the $\mathscr{G}_s$ that folds its identical neighbors in the meta-paths, $\wedge$ is the linearization-based concatenation operator, π checks the amount of information transmitted via p and $\bar{n}_\ell$ is the 1-hop target node that is connected via the NSP modeling technique.

To achieve the *first*, *second* and *higher* order path connection hierarchies at any stage, we maintained the sequential inter-triplet path-flow by separately defining the linearization expressions for any two consecutive nodes of each and every meta-path. Finally, we concatenated the expressions of the consecutive paths to bridge k-order linear path-flow (i.e., meta-paths-based information flow).

To improve the readability of the linearization process, we term n_c as n_k and n_ℓ as n_{k-1} to describe the objective function of 1-order path linearization as:

$$\bar{n}_{\mathcal{N}(n_k)} = \bigwedge_{n_k,\ p,\ n_{k-1}\in\mathcal{N}(n_k)} \pi(n_k,\ p,\ n_{k-1}) \vdash \mathcal{I}_{k-1}\cdot \bar{n}_{k-1} \tag{16}$$

The objective function of 2-order path linearization is defined as:

$$\bar{n}_{\mathcal{N}(\mathcal{N}(n_{k-1}))} = \bigwedge_{n_{k-1},\ p,\ n_{k-2}\in\mathcal{N}(n_{k-1})} \pi(n_{k-1},\ p,\ n_{k-2}) \vdash \mathcal{I}_{k-2}\cdot \bar{n}_{k-2} \tag{17}$$

$$\vdots$$

Similarly, the objective function of i-order path linearization is defined as:

$$\bar{n}_{\mathcal{N}^i\left(n_{k-(i-1)}\right)} = \bigwedge_{n_{k-(i-1)},\ p,\ n_{k-i}\in\mathcal{N}\left(n_{k-(i-1)}\right)} \pi\left(n_{k-(i-1)},\ p,\ n_{k-i}\right) \vdash \mathcal{I}_{k-i}\cdot \bar{n}_{k-i} \tag{18}$$

where i is fixed in the range of 1 to 5 for a feasible conduct of the experimental work.

Algorithm 2. Interactions-enhanced Knowledge propagation Network (IKN)

Inputs: Subgraph: $\mathcal{G}_s(E,\mathcal{P})$, Interaction Matrix: $\mathcal{M}$, Neighborhood sampling: $K = 2^i$, Hyper-parameters: K, d, s Training Parameters: $n_e \in E, p_i \in P$ & Hidden Units.

Output: Preference Prediction $\hat{\delta}_{(u,v)} = \varphi\left[(n_u, n_v) \otimes \left(\mathcal{G}_s, \mathcal{M}_{uv}, \Theta, \bar{\boldsymbol{h}}_{uv}(n_u, n_v)\right)\right]$

Initialize the required parameters;
Pre-train SSC module with Algorithm 1
Calculate hop-set $\left\{n^{(s)\frown(s-k)}_{(n_k)} \vdash \mathcal{I}_k\right\}_{k=1}^{K}$ for each user;
do
    foreach (t) in $\mathcal{M}$ **do**
        Sample minibatch Ω of + ve and –ve iterations from $\mathcal{M}$;
        Sample any entity e wrt $n^{(s)\frown(s-k)}_{(n_k)} \vdash \mathcal{I}_k$ for each item i in Ω.
        Update $\hat{\delta}_{(u,v)}$ via gradient descent;
    end foreach
    Sample minibatch of $A(n_c, p, n_\ell) \in \mathcal{G}_s$ & $B(a, p, b) \notin \mathcal{G}_s$ from $\mathcal{G}_s$ as in Eq. (36).
    Calculate forfeiture $\mathcal{F}$ wrt $\mathcal{U}, \mathcal{V}, E, \mathcal{R}$ of each gradient on Ω.
    Use gradient descent to update $\mathcal{U}, \mathcal{V}, E, \mathcal{R}$ with learning rate η.
while (Training Iterations)! converge
Return $\hat{\delta}_{(u,v)}$

4.4. Feedback-attributed interactions Enhancement (FIE)

FIE exploits GNN to learn (i.e., extract and preserve) higher-order KGE-based explicit feedback features from the underlying data. The features are used to simultaneously augment user-item interactions and enhance system's regularization capability to highlight the influential nodes in IKN. Typically, the embedded representations of users, items, paths, nodes and interactions (i.e., $\mathcal{U}$, $\mathcal{V}$, $\mathcal{P}$, E and $\mathcal{I}_\ell$ respectively) are received from the projection layer and integrated into the generalization layer-based Activatable Operands Pool (AOP) $\mathscr{J}$ = $\left[\mathcal{U}, \mathcal{V}, \mathcal{P}, n_h, n_t, \mathcal{I}_\ell, c^{n_i}_{in}, c^{n_i}_{out}\right]$ with $\mathscr{J}^{(0)}$ as the initial integration, where $n_h, n_t \in E$ are the head & tail nodes respectively, and c is the count of the number of incoming and outgoing relations to/from a specific node i. We applied Swish activation to activate $\mathscr{J}$ as:

$$\mathscr{J}^{(k+1)} = f(x\cdot\sigma(x)) \tag{19}$$

where k is 0 initially, $\mathscr{J}^{(k+1)}$ is the activated AOP, $f(x) = \mathcal{W}\mathscr{J}^{(k)}\cdot\mathcal{M}_{uv} + b$, $\mathcal{W}$ is the weight matrix, b is a bias factor of the k th layer, and $\mathcal{M}_{uv}$ is an interaction matrix containing u-v interactions.

Generally, the features' extraction process faces a huge relevancy loss since the extracted personalized features partially conform to the underlying interactions. Relevant features are exploited to enhance the interactions and the irrelevant data is considered as the generalization forfeiture (i.e., loss). Concerning the given AOP, only the relevant features belong to $\mathscr{J}^{(k)}$, therefore, the generalization forfeiture is defined as:

$$\mathscr{F}_{\mathrm{FIF}} = \sum_{(\gamma_i)\notin \mathscr{J}^{(k)}} \log\left[1 - f(x_\gamma \cdot \sigma(x_\gamma))\right] - \sum_{(\gamma_i)\in \mathscr{J}^{(k)}} \log\left[f(x_\gamma \cdot \sigma(x_\gamma))\right] + \lambda\|\Theta\|_2^2 \tag{20}$$

where γ is the features' index, $f(x) = \mathrm{W}_{\mathscr{J}^{(k)}} + b$, λ is the ENR[11] and Θ is the set of model parameters.

We used Adam optimizer to optimize the forfeiture and steer the learning rate according to the absolute value of the corresponding gradient. We adjusted the learning parameters through back propagation by maximizing the log-likelihood of $(\gamma_i) \in \mathscr{J}^{(k)}$ and defined the learning process as:

$$\Upsilon_{learn} = \sum_{l=1}^{L} \sum_{(\gamma_i)\in \mathscr{J}^{(k)}} \log \Pi(f(x_\gamma \cdot \sigma(x_\gamma))) \tag{21}$$

where $\Pi(f(x_\gamma \cdot \sigma(x_\gamma))) = \frac{\exp(f(x_\gamma \cdot \sigma(x_\gamma)))}{\sum_{\forall \gamma_j \in \mathscr{J}^{(k)}} \exp\left(f\left(x_{\gamma_j} \cdot \sigma\left(x_{\gamma_j}\right)\right)\right)}$ is the softmax function.

4.5. Sampling, retrieval and formalization (SRF)

We innovatively used *Inverse Square Law*[12] to sample the influential nodes up to a p-length of 5 hops in IKN and independently moved their referential information to the LCN (i.e., n_c) of the current community as shown in the Fig. 4. The operation defined to move the preferences from $n_{\mathcal{N}(n_c)}$ to n_c wrt s is $n_c^{(s)} = f(n_c, n_{\mathcal{N}(n_c)})$, where s is the path-step ranging from 1 to 5, and $f(\cdot)$ is a function used to directly move the preferences from source to destination. A non-linear transformation is applied on the output of the move-operation to automatically adjust the moved preferences in the LCN. Prior the move operations, we applied softmax normalizer to normalize the attention scores of the coefficients of influential candidates in $\mathcal{N}(n_c)$ as:

$$\pi(n_c, p, n_\ell) \vdash \mathscr{I}_\ell = \frac{\exp(\pi(n_c, p, n_\ell) \vdash \mathscr{I}_\ell)}{\sum_{n_i \in \mathcal{N}(n_c)} \exp(\pi(n_c, p, n_i) \vdash \mathscr{I}_i)} \tag{22}$$

The move operation is augmented via the trainable weight matrices and activated via the activation function as:

$$f_{move} = f(n_c, n_{\mathcal{N}(n_c)}) = \Phi(\mathscr{W} \cdot (n_c \leftarrow n_{\mathcal{N}(n_c)} + b) \tag{23}$$

where Φ is the Swish – a non-linear activation function, and b is a bias factor, $\mathscr{W} | \mathscr{W} \in \mathbb{R}^{d^{\mathscr{T}} \times k}$ is the trainable weight matrix, $d^{\mathscr{T}}$ is the transformation size, and $d^{\mathscr{T}} \times k$ is the transformation size of the k th layer.

For the generalized description of a meta-path-based move operation, we recursively extended Eq. (23) according to Eq. (15) to express higher-order path connections as:

$$n_{(n_k)}^{(s)\frown(s-1)} = f\left(n_{(n_k)}^{(s)}, n_{\mathcal{N}(n_{k-1})}^{(s-1)}\right) | n_k = c \tag{24}$$

$$n_{\mathcal{N}(n_{k-1})}^{(s-1)\frown(s-2)} = f\left(n_{\mathcal{N}(n_{k-1})}^{(s-1)}, n_{\mathcal{N}(\mathcal{N}(n_{k-2}))}^{(s-2)}\right) \tag{25}$$

$$n_{\mathcal{N}(\mathcal{N}(n_{k-2}))}^{(s-2)\frown(s-3)} = f\left(n_{\mathcal{N}(\mathcal{N}(n_{k-2}))}^{(s-2)}, n_{\mathcal{N}(\mathcal{N}(\mathcal{N}(n_{k-3})))}^{(s-3)}\right) \tag{26}$$

$$\vdots$$

$$n_{\mathcal{N}^i(n_{k-i})}^{(s-i)\frown(s-(i+1))} = f\left(n_{\mathcal{N}^i(n_{k-i})}^{(s-i)}, n_{\mathcal{N}^{i+1}\left(n_{(k-(i+1))}^{(s-(i+1))}\right)}\right) \tag{27}$$

where s shows the hop-length (i.e., path step) and $i = 1$ to 5.

To linearize move operations wrt a k-order path and interaction hierarchy (i.e., the k-hop meta-path to be initiated from the 1-hop), we adjusted Eq. (24) according to Eq. (16) as:

$$n_{\mathcal{N}(n_k)}^{(s)\frown(s-i)} = \bigwedge_{\mathscr{O}_p^{(s)}} \pi(n_k, p, n_{k-1}) \vdash \mathscr{I}_{k-1} \cdot n_{k-1} \| \bigwedge_{\mathscr{O}_p^{(s-i)}} \pi(n_{k-i}, p, n_{k-(i+1)}) \vdash \mathscr{I}_{k-(i+1)} \cdot n_{k-(i+1)} \tag{28}$$

where $\mathscr{O}_p^{(s)} = n_k, p, n_{k-1} \in n_{\mathcal{N}(n_k)}$, $\mathscr{O}_p^{(s-i)} = n_{k-i}, p, n_{k-(i+1)} \in n_{\mathcal{N}^i(n_{k-1})}$ and $\mathscr{O}$ describes the p length.

Finally, the phenomenon is generalized as:

[11] Elastic Net Regularization (ENR) is the combination of (0 for) ridge and (1 for) lasso coefficients, i.e., ℓ_1 and ℓ_2 regularizations respectively, to get optimal prediction results. Through applying ENR, we can select optimal values between 0 and 1 to expand/shrink the coefficients to boost the performance and/or assign 0s to all of the sparse-coefficients to avoid the data-sparsity.

[12] The strength of the information of a *meta*-path is inversely proportional to the square of the length of the *meta*-path.

$$n^{(s)\backsim(s-i)}_{\mathscr{N}(n_k)} = n^{(s)}_{(n_k)} \parallel n^{(s-i)}_{\mathscr{N}^i(n_{k-i})} \tag{29}$$

where $n^{(s)}_{(n_k)}$ represents the n_c with hop-distance s (i.e., the central node), $n^{(s-1)}_{(n_k)}$ shows the next node with hop-distance $s-1$ (i.e., 1-hop neighbor), and so on. Similarly, $n^{(s)\backsim(s-1)}_{(n_k)}$ can be termed as $n^{(1)}_c$ since it is the central node connected to its 1-hop neighbor through the physical path-superscript of $(s)\backsim(s-1)$, and so on. The superscript $(s)\backsim(s-1)$ is equal to $(0)\backsim(0 \text{ to } 1) \cong (1)$ and $\mathscr{N}$ shows the set of neighbors of the concerned node. Hence, the required i-hop lengthy meta-paths from n_c are maintained and node-to-node linearized to move back the preferential information to LCN, and i can have any value in the range of 1 to 5 wrt the meta-paths.

To independently collect the preferential information from different LCN, we applied the gated formalization technique to streamline the information from all LCN to the main unit. At this stage, in IKN, only the LCN are considered as the influential nodes and the rest are overlooked. Specifically, we used the reset (r) gate and the update (u) gate through the Swish function – a self-gated mechanism – to formalize the preferences as:

$$r^s_{n_k} = \mathscr{W}_r(x) \cdot \sigma(\mathscr{W}_r(x)) \tag{30}$$

$$u^s_{n_k} = \mathscr{W}_u(x) \cdot \sigma(\mathscr{W}_u(x)) \tag{31}$$

where $\mathscr{W}_r$ and $\mathscr{W}_u$ are the weight matrices wrt to the r and u gates, and $x = n^s_{n_k} + h\left(n^s_{n_k}\right), n^{s-1}_{n_k-1} + h\left(n^{s-1}_{n_k-1}\right)$, thus, the candidate hidden state $\bar{h}$ of n_c is comprehensively defined as:

$$\bar{h}\left(n^s_{n_k}\right) = \Phi\left[\mathscr{W}_{\mathrm{h}}\left(\mathrm{n}^s_{\mathrm{n_k}} + \mathrm{r}^s_{\mathrm{n_k}}\left(\left(\mathrm{n}^s_{\mathrm{n_k}} + \mathrm{h}\left(\mathrm{n}^s_{\mathrm{n_k}}\right)\right) \odot \left(\mathrm{n}^{s-1}_{\mathrm{n_k}-1} + \mathrm{h}\left(\mathrm{n}^{s-1}_{\mathrm{n_k}-1}\right)\right)\right)\right)\right] \tag{32}$$

where $\bar{h}$ shows the candidate hidden state which is comprised of the current and neighbor states as well as the current and neighbor hidden states, $r^s_{n_k}$ is the r-gate that observes information loss in $h\left(n^s_{n_k}\right)$ and $h\left(n^{s-1}_{n_k-1}\right)$, and $\mathscr{W}_h$ is a weight matrix that contains information about the coefficients of the hidden states.

Similarly, the current hidden $\bar{\boldsymbol{h}}_\ell$ state of n_c, i.e., $\bar{\boldsymbol{h}}_\ell(n_c) = \bar{\boldsymbol{h}}_\ell\left(n^s_{n_k}\right)$, is defined as:

$$\bar{\boldsymbol{h}}_t\left(n^s_{n_k}\right) = \left[\left(1 - u^s_{n_k}\right) \cdot \left(\left(n^s_{n_k} + h\left(n^s_{n_k}\right)\right) \odot \left(n^{s-1}_{n_k-1} + h\left(n^{s-1}_{n_k-1}\right)\right)\right] + \left(u^s_{n_k} \odot \bar{\mathrm{h}}\left(n^s_{n_k}\right)\right) \tag{33}$$

where $\bar{\boldsymbol{h}}_\ell\left(n^s_{n_k}\right)$ shows the current hidden state which is comprised of u-gate-based current and previous states, the current and previous hidden states, and u-gate influenced candidate hidden state.

In the k-order path connections, Eq. (32) is recursively called to update the respective representations of each next node at the upper steps. For instance, $\bar{\boldsymbol{h}}_\ell\left(n^s_{n_k}\right)$ independently formalizes the information of current and previous nodes, current and previous hidden nodes, k-hop neighbor nodes, k-hop neighbor hidden nodes, and the corresponding relations along the travelling path at k th step to the destination.

4.6. Preference prediction

This is the main prediction unit that calculates the finalized preferences and generates the recommendations. Specifically, the "*Predictor*" receives meta-path-based k-order entity representations and hidden information of the concerned influential nodes from SRF module to predict the required preferences as:

$$\widehat{\delta}_{(u,v)} = \varphi\left[(n_u, n_v) \otimes \left(\mathscr{G}_s, \mathscr{M}_{uv}, \Theta, \bar{\boldsymbol{h}}_{uv}(n_u, n_v)\right)\right] \tag{34}$$

where $\widehat{\delta}_{(u,v)}$ is the recommendation list, φ is the multilayer perceptron, $\otimes$ denotes the model's binding operator to rationalize the projection, $\mathscr{M}$ represents the interaction matrix, n_u and n_v are the representations of users and items respectively, Θ shows the model parameters and $\bar{\boldsymbol{h}}_{uv}$ represents the concerned hidden units.

The concatenated embeddings of u-v interactions are provided to φ to model the interactions. An equally divide-&-conquer strategy is adopted on each higher layer by exploiting the hidden units to model the abstracted information. Moreover, we applied Swish activation function to activate the hidden layers and utilized sigmoid function on the top of the output layers to restrict (i.e., normalize) the outcomes in a range of 0 to 1.

4.7. Recommendation explainability

The questions "*why an individual should accept, believe or trust the provided recommendations?*", "*why an item is being recommended to a specific person?*" or "*how a RS can fulfil the necessities of the concerned users?*" are rationally responded by the explainability of the provided recommendations. Recommendation depends on user interactions with the items such as view, like, dislike, comment, purchase, rate, etc. on different online platforms like e-commerce websites, social networks, advertisement circles, etc., and so does the explainability of recommendation [25]. Typically, SAGE exploits similarity-enhanced meta-paths to recommend unseen items to the end users based on the highest similarity of the items with the interaction history of the users via the attention scores. For instance, the interest of *"Ahmad"* in the book of *"Programming"* can be described via the meta-path *"Ahmad* $\stackrel{studied}{\rightarrow}$ *OOP in C++* $\stackrel{is-written-by}{\rightarrow}$ *Robert Lafore* $\stackrel{is-author-in}{\rightarrow}$ *Programming*"; if the node *"Programming"* had the highest similarity attention score with *"OOP in C++"*, and *"Ahmad"* had 1-hop and 2-hop attentive distances with *"OOP in C++"* and *"Robert Lafore"* respectively. The explainability emphasis between nodes varies with the length of path between the given nodes [12,25]. Therefore, we explain how meta-paths are tackled by the proposed approach wrt the tradeoff between explainability emphasis and hop-length. For instance, a meta-path $\wp_j$ *"Ahmad* $\stackrel{studied}{\rightarrow}$ *ALP for IBM* $\stackrel{is-written-by}{\rightarrow}$ *Robert Lafore* $\stackrel{is-author-in}{\rightarrow}$ *Programming* $\stackrel{includes}{\rightarrow}$ *OOP with Java*" is translated to $\rho_j = n_1 \stackrel{p_1}{\rightarrow} n_5 \stackrel{p_5}{\rightarrow} n_8 \stackrel{p_9}{\rightarrow} n_7 \stackrel{p_7}{\rightarrow} n_{10}$ for an easier manipulation as shown in Table 1. We enlighten the translation of meta-paths (i.e., subgraph) to ρ-based flow diagram and adjacency matrix as shown in Fig. 5 and Table 1(a) respectively.

Contrasting with the exploitation of manually tailored path patterns and random walk-based strategies [9,23], SAGE automatically captures highly relevant trajectories via NSP modeling for recommendation. The meta-paths with relevant information remarkably alleviate the tradeoff between explainability emphasis and hop-length. Moreover, we demonstrate a visualized case study in Section 5.6 to systematically describe the explainability of the proposed approach.

4.8. Optimization

Decisions depend on the information in hand. The interactions observed among users and items are 1 s and 0 s otherwise. The observed interactions contribute to preference acquisition only and 0 s are deemed as loss. Thus, the forfeiture (current loss) of the testimony of optimal prediction is defined via the following expression:

$$\mathscr{F}_{OP} = \sum_{(u_i,v_j)\in\mathscr{I}^0} \log\left[1 - \left(\widehat{\delta}_{(u_i,v_j)} \cdot \sigma\left(\widehat{\delta}_{(u_i,v_j)}\right)\right)\right] - \sum_{(u_i,v_k)\in\mathscr{I}^1} \log\left[\widehat{\delta}_{(u_i,v_k)} \cdot \sigma\left(\widehat{\delta}_{(u_i,v_k)}\right)\right] \tag{35}$$

where $\mathscr{F}_{op}$ shows the forfeiture of the testimony of optimal prediction, and $\mathscr{I}^0$ and $\mathscr{I}^1$ represent the un-observed and observed interactions respectively.

The overall forfeiture of SAGE depends on four aspects of implementation, i.e., (i) translation $\mathscr{F}_{Trans}$, (ii) similarity calculation $\mathscr{F}_{Sim}$, (iii) FIF $\mathscr{F}_{FIF}$, and (iv) Prediction $\mathscr{F}_{op}$.

Therefore, for (i), the forfeiture is defined as:

$$\mathscr{F}_{\mathrm{Trans}} = \sum_{\forall \mathrm{B}\notin\bar{\mathrm{G}}} \log\ \left[1 - \left(\mathscr{g}(\mathrm{B}) \cdot \sigma_{\mathscr{g}}(\mathrm{B})\right)\right] - \sum_{\forall \mathrm{A}\in\bar{\mathrm{G}}} \log\ \left[\mathscr{g}(\mathrm{A}) \cdot \sigma_{\mathscr{g}}(\mathrm{A})\right] \tag{36}$$

where $B|B = (a, p, b)$ shows an invalid triplet, a and b represent e_h and e_t respectively but their data is either missing or unreadable, and $A|A = (n_c, p, n_\ell)$ is a valid triplet.

For (ii), irrelevance ratio is considered as the forfeiture that is defined as:

$$\mathscr{F}_{Sim} = \sum_{\forall Y\notin\mathscr{G}} \log\left[1 - (Sim(Y) \cdot \sigma Sim(Y))\right] - \sum_{\forall Z\in\mathscr{G}} \log\left[Sim(Z) \cdot \sigma Sim(Z)\right] \tag{37}$$

where $Y|Y = (a, b)$ and $Z|Z = (n_c, n_\ell)$ represent the irrelevant and relevant nodes respectively.

Similarly, for (iii) and (iv), the required forfeitures (i.e., $\mathscr{F}_{FIF}$ and $\mathscr{F}_{OP}$) are defined in Eq. (20) and Eq. (35) respectively.

Therefore, $\mathscr{F}_{SAGE} = \mathscr{F}_{Trans} + \mathscr{F}_{Sim} + \mathscr{F}_{FIF} + \eta\mathscr{F}_{op} + \lambda\|\Theta\|_2^2$ is the total forfeiture of the proposed approach, where η is a hyper parameter used to counter-balance the effect of overall forfeiture at the model's learning, λ shows the ENR weight, and Θ represents the set of model parameters.

We optimized $\mathscr{F}_{SAGE}$ via SGD[13] algorithm that controls the training rate according to the absolute value of the corresponding gradient. We computed the forfeiture gradients wrt Θ and updated the parameters by back propagation through the minibatch of random samples and maximizing the log-likelihood of $(u_i, v_k) \in \mathscr{I}^1$. Therefore, the Υ_{train} objective function and the optimal training forfeiture is defined as:

$$\Upsilon_{train} = \sum_{s=1}^{S} \sum_{(u_i,v_k)\in\mathscr{I}^1} \log \Pi\left(\widehat{\delta}_{(u_i,v_k)} \cdot \sigma\left(\widehat{\delta}_{(u_i,v_k)}\right)\right) \tag{38}$$

Where $\Pi\left(\widehat{\delta}_{(u_i,v_k)} \cdot \sigma\left(\widehat{\delta}_{(u_i,v_k)}\right)\right) = \frac{\exp\left(\widehat{\delta}_{(u_i,v_k)} \cdot \sigma\left(\widehat{\delta}_{(u_i,v_k)}\right)\right)}{\sum_{\forall(u_i,v_K)\in\mathscr{I}^1} \exp\left(\widehat{\delta}_{(u_i,v_K)} \cdot \sigma\left(\widehat{\delta}_{(u_i,v_K)}\right)\right)}$ shows the softmax function.

[13] Stochastic Gradient Descent.

Table 1
Matrix-based representation of nodes, paths, meta-paths and their mutual low-level control flow. The description of the nodes and paths used in Table 1(a) is provided in Table 1(b).

Table 1**(a).** Adjacency Matrix $\mathscr{A}$ based on the meta-Paths.

	$\mathbf{n_1}$	$\mathbf{n_2}$	$\mathbf{n_3}$	$\mathbf{n_4}$	$\mathbf{n_5}$	$\mathbf{n_6}$	$\mathbf{n_7}$	$\mathbf{n_8}$	$\mathbf{n_9}$	$\mathbf{n_{10}}$	$\mathbf{n_{11}}$
$\mathbf{n_1}$	0	p_2	0	p_1	p_1	0	0	0	0	0	0
$\mathbf{n_2}$	0	0	p_1	0	0	0	0	0	0	0	0
$\mathbf{n_3}$	0	0	0	0	0	p_3	p_4	0	0	0	0
$\mathbf{n_4}$	0	0	0	0	0	0	p_4	p_5	0	0	0
$\mathbf{n_5}$	0	0	0	0	0	0	0	p_5	0	0	0
$\mathbf{n_6}$	0	0	0	0	0	0	0	0	p_6	0	0
$\mathbf{n_7}$	0	0	0	0	0	0	0	0	0	p_7	0
$\mathbf{n_8}$	0	0	0	0	0	0	p_9	0	0	0	p_8
$\mathbf{n_9}$	0	0	0	0	0	0	p_{10}	0	0	0	0
$\mathbf{n_{10}}$	0	0	0	0	0	0	0	0	0	0	0
$\mathbf{n_{11}}$	0	0	0	0	0	0	0	0	0	0	0

Table 1**(b).** Description of the Nodes and Paths used in the Adjacency Matrix $\mathscr{A}$ of the Table 1(a).

Nodes	**Description**	**Paths**	**Description**
$\mathbf{n_1}$, $\mathbf{n_2}$	Ahmad, Asad	$\mathbf{p_1}$	studied
$\mathbf{n_3}$	H/F Java	$\mathbf{p_2}$	is-friend-of
$\mathbf{n_4}$	OOP in C++	$\mathbf{p_3}$	is-published-by
$\mathbf{n_5}$	ALP for IBM	$\mathbf{p_4}$	belongs-to
$\mathbf{n_6}$	Publisher	$\mathbf{p_5}$	is-written-by
$\mathbf{n_7}$	Programming	$\mathbf{p_6}$	is-publisher-of
$\mathbf{n_8}$	Robert Lafore	$\mathbf{p_7}$	includes
$\mathbf{n_9}$	H/F JavaScript	$\mathbf{p_8}$	is-author-of
$\mathbf{n_{10}}$	OOP with Java	$\mathbf{p_9}$	is-author-in
$\mathbf{n_{11}}$	DS & Algo	$\mathbf{p_{10}}$	supports

$$\min_{train} \mathscr{F} = \sum_{(u_i, v_k) \in E} \mathscr{F}\left(\bar{Y}_{uv}, Y_{uv}\right) + \frac{\varepsilon}{2} \sum_{p \in \mathscr{P}} \|\zeta_p - E^{\top} \mathscr{P} E\|_2^2 + \frac{\lambda}{2}\left(\|X\|_2^2 + \sum_{p \in \mathscr{P}} \|\mathscr{P}\|_2^2\right) \tag{39}$$

where $\mathscr{F}\left(\bar{Y}_{uv}, Y_{uv}\right) = -[\mathscr{M}_{uv} log(u^{\top} v \cdot \sigma(u^{\top} v)) + (1 - \mathscr{M}_{uv}) \log(1 - (u^{\top} v \cdot \sigma(u^{\top} v)))]$ is used to calculate the cross-entropy forfeiture between the predicted and the true interactions. The second part of Eq. (39) computes the square-error between the true and the prediction-based reconstructed indicator $E^{\top}\mathscr{P}E$ matrix, and the last part indicates the overfitting regularization process. Moreover, $x \cdot \sigma(x)$ shows the operations of swish activator, E and $\mathscr{P}$ represent the embedding-sets of entities and paths respectively, ζ_p denotes the representation of indicator tensor ζ for path p in KG, and X contains $\{\mathscr{U}, \mathscr{V}, E, \mathscr{W}\}$.

The computational Time Complexity (TC) depends on four different aspects of SAGE, i.e., (i) Translation, (ii) SSC, (iii) IKN and Prediction, (iv) FIE and SRF. For (i), $O\left(\left|\bar{G}\right| d^2\right)$ is the TC of translation, where $\bar{G}$ shows the graph. For (ii), the TC of node similarity is $O\left(|TNP| d^2\right)$, and the TC of semantic similarity is $O(|Sem_{Sim}| d)$. Paths and meta-paths are internally combined to access the information. Therefore, $O(|pN| d + |plogN| d)$ and $O(|\wp M| d + |\wp logM| d)$ are the TCs of integration of paths into the meta-paths and meta-paths into the graph respectively. The total TC of (ii) is $O\left(|TNP| d^2 + |Sem_{Sim}| d + |pN + \wp M| d + |plogN + \wp logM| d\right) \simeq O\left(|n_{Sim}| d^2 + |C| d + |logC| d\right)$, where p is the path, N is the number of paths in meta-path, $\wp$ is the meta-path, M is the number of meta-paths in graph, and C is the constant. For (iii), $O\left(\sum_{i=1}^{k} \left|\bar{G}\right| d_i d_{i-1}\right)$ is the TC of matrix processing at k-order propagation, and $O\left(\sum_{i=1}^{k} \left|\bar{G}\right| d_i\right)$ is the TC of inner product of k-epochs at prediction, where d is the embedding size. Thus, the total TC of (iii) is $O\left(\sum_{i=1}^{k} \left|\bar{G}_i\right| d_i d_{i-1} + \sum_{i=1}^{k} \left|\bar{G}_i\right| d_i\right) \simeq \sum_{i=1}^{k} O\left(\left|\bar{G}_k\right| d_k^2\right)$. For (iv), the TC is linear. By adding the resultant TCs from all of the four parts, we obtained $O\left(\left|\bar{G}\right| d^2 + |n_{Sim}| d^2 + |C| d + |\log C| d + \sum_{i=1}^{k} \left|\bar{G}_k\right| d_k^2\right) \approx \sum_{s=1}^{S} O\left(\left|\bar{G}_s\right| d_s^2\right)$, where s shows the hop-length and $s \in S$. Therefore, the general computational TC of SAGE is declared as $O\left(|\mathscr{I}^1| \sum_{s=1}^{S} \left|\bar{G}_s\right| d_s^2\right)$ that is quadratic in calculation of similarity, translation and propagation, and linear in the rest of the computation.

5. Experiments

In this section, we intend to satisfactorily answer the following research questions.

RQ1: What is the performance of SAGE against the-state-of-the-art models in KG-based recommendations?
RQ2: What is the impact of different modules on the performance of SAGE?
RQ3: What is the impact of different adjustments of hyper-parameters on the performance of SAGE?
RQ4: What is the performance of SAGE against cold start and data sparsity problems?
RQ5: Can SAGE generate explainable recommendations?

5.1. Experimental setup

We discuss experimental process and results' analysis in this section in details.

5.1.1. Benchmark datasets

We performed full-fledged experiments on three real-world datasets, i.e., Book-Crossing, Last-FM and Yelp to assess the effectiveness of SAGE. Specifically, **Book-Crossing**[14] contains incisive ratings from different users about numerous books in a range of 0 to 10. It is basically collected from the community-web of BookCrossing.[15] **Last-FM**[16] contains users, artists and past listening records of the users about the performance of different musicians. Mainly, this dataset is collected from the Last.fm online system where the music tracks are considered as the items. **Yelp**[17] contains user ratings ranged from 1 to 5 about items such as restaurants, foods, bars, etc. It also covers various features of items and social relations among users. This version is collected from Yelp-challenge's edition published in 2018.

We used 1 as a sign of occurrence of interactions between user and book, used ID embeddings of entities as the preliminary inputs during the pre-processing of *Book-Crossing* dataset, and selected relevant information from MS-Satori to enrich it. We kept RCV[18] greater than 0.8 to preserve health of the relations among entities during the extraction of triplets having any match of the word "book" in their information tags using a formatted query (i.e., $e_h, *.book.*, e_t$) of graph crawling, where e_h and e_t are also checked for a possibility of the word "book". We discarded the redundant and invalid triplets and synchronized e_t and e_h of the current and new triplets respectively wrt their IDs in the underlying information. We further extended this process to four hops wrt each current triplet and five hops wrt each individual node. Moreover, we mapped the items of *Last-FM* to the publically available mappings of DBpedia to enrich them according to [40]. We retrieved map-able triplets from DBpedia ontology[19] with a relation frequency greater than five and having direct connections to the entities aligned with items. In case of *Yelp*, we extracted items' information such as category, features, location, etc. from different business-based networks to enrich low level representations of the items. We preprocessed the extracted information to eliminate redundant, missing or less frequently occurring data items. Formally, we normalized the datasets by removing users with less than 5 ratings and items having no conforming entities in the external KGs. We divided the datasets in Training Y_{train}, Testing Y_{test} and Validation Y_{val} parts where each contains 70, 20 and 10 percent of the given ratings respectively as summarized in Table 2. Finally, we termed this data as the SAGE-Dataset,[20] released it and published in the Mendeley Data.

5.1.2. Amazon customer review dataset (ACRD) – Online Feeds

We have assessed the proposed model on ACRD[21] – Online Feeds that are accessible through the online industrial recommendation application interface of Amazon (i.e., Amazon Personalize[22]). ACRD contains 30 M + customer views (i.e., implicit interactions) and 100 K plus customer ratings (i.e., explicit interactions) of over 5 M users about over 2 M items as summarized by SNAP.[23] We preprocessed the data as the above mentioned public datasets, obtained 8027 entities, 13,572 unique interactions, 10 path-types and 23,984 path-counts, and constructed the local subgraph for experiments in the same way. The interactions observed (i.e., either implicit or explicit) are deemed as 1 s and 0 s otherwise.

5.1.3. Evaluation

We selected five commonly used evaluation-metrics (i.e., AUC, Accuracy, Precision, Recall and NDCG) to evaluate the effectiveness of the proposed approach in comparison with the baseline methods.

AUC[24] is the ratio of the likelihood that an arbitrarily irrelevant-item is ranked lower than an arbitrarily relevant-item. It is the measure of the successful recommendations based on the comparison of ROC curves of different models through the logistic regression defined as:

[14] https://www2.informatik.uni-freiburg.de/~cziegler/BX/.
[15] https://www.bookcrossing.com/.
[16] https://files.grouplens.org/datasets/hetrec2011/hetrec2011-lastfm-readme.txt.
[17] https://www.yelp.com/dataset/challenge.
[18] Relation Confidence Value.
[19] https://wiki.dbpedia.org/services-resources/ontology.
[20] http://dx.https://doi.org/10.17632/ky72sb6w6h.1.
[21] https://s3.amazonaws.com/amazon-reviews-pds/readme.html.
[22] https://aws.amazon.com/personalize/.
[23] https://snap.stanford.edu/data/web-Amazon.html.
[24] Area under the ROC (Receiver Operating Characteristic) Curve.

Table 2
Comprehensive statistics of the utilized datasets.

Literals	Datasets		
	Book-Crossing	**Last-FM**	**Yelp**
Domain	Books	Music	Pol[27]
Users u	16,878	1865	45,909
Items i	13,272	6526	1358
Unique Interactions $\mathscr{I}$	120,335	68,456	1,185,068
Nodes	75,253	12,039	90,961
Path-Types	23	58	32
Path-Counts	263,694	29,650	1,853,704
Training	84,235	47,920	829,548
Testing	24,067	13,692	237,014
Validation	12,034	4846	118,507

[27] Point of Interest.

$$AUC = \frac{1}{|\mathscr{U}|}\sum_{u_i}\frac{1}{|P(u_i)|}\sum\nolimits_{(v_k,v_l)\in P(u_i)}\xi\left(\widehat{\delta}_{(u_i,v_k)} > \widehat{\delta}_{(u_i,v_l)}\right) \tag{40}$$

where P shows the evaluation-pairs based on the types of items wrt a specific user and P is defined as:

$$P(u_i) = \{(v_k, v_l|(u_i, v_k) \in Y_{test} \wedge (u_i, v_l) \notin (Y_{test} \cup Y_{train} \cup Y_{val})\} \tag{41}$$

where v_k and v_l are the interaction-enhanced items and un-enhanced items respectively wrt a specific user u_i, and $\xi(\cdot)$ is a function used to return higher index of the interaction-enhanced items over the un-enhanced items.

Accuracy is the ratio of correctly recommended items to the total items defined as:

$$\text{Accuracy} = \frac{\sum_{v_k\in\mathscr{V}}|Rec(v_k)\cap CorRec(v_k)|}{\sum_{v_k\in\mathscr{V}}|Rel(v_k)\cup Rec(v_k)\cup CorRec(v_k)|} \tag{42}$$

Precision is the ratio of correctly recommended items to the total recommended items defined as:

$$\text{Precision} = \frac{\sum_{v_k\in\mathscr{V}}|Rec(v_k)\cap CorRec(v_k)|}{\sum_{v_k\in\mathscr{V}}|Rec(v_k)|} \tag{43}$$

Recall is the ratio of correctly recommended items to the total relevant items defined as:

$$\text{Recall} = \frac{\sum_{v_k\in\mathscr{V}}|Rec(v_k)\cap CorRec(v_k)|}{\sum_{v_k\in\mathscr{V}}|Rel(v_k)|} \tag{44}$$

NDCG[25] is the ratio of the ranking quality of top-K recommendation based on the applied standards to measure the efficiency of results through ranking the discounted significance defined as:

$$NDCG = \sum_{i=1}^{|Rec(v_k)_n|}\frac{2^{Rec(v_k)_n}-1}{\log_2(i+1)}\cdot\left(\sum_{i=1}^{|CorRec(v_k)_n|}\frac{2^{CorRec(v_k)_n}-1}{\log_2(i+1)}\right)^{-1} \tag{45}$$

where $Rec(v_k)$, $CorRec(v_k)$ and $Rel(v_k)$ are the recommended, correctly recommended and relevant items respectively. Higher are the outcomes of these metrics, higher is the quality of recommendations and vice versa.

For offline preference prediction, we evaluated the models in two steps, i.e., (a) via CTR[26] prediction, and (b) via top-K recommendation. In (a), we acquired the probability of the predicted clicks via the application of trained model on each interaction in the testing set and used AUC and Accuracy for evaluation. In (b), we acquired the probability of the predicted clicks via the application of trained model against each user in the testing set and utilized (Precision, Recall, NDCG) @K for performance evaluation.

For online performance evaluation, wrt the items' appearance in recommendation responses, we assessed SAGE and the state-of-the-art models based on *"Item Appearance in General Recommendation-response to the Query*" (IAGRQ) and *"Item Appearance in Special Recommendation-response to the User*" (IASRU) via CTR prediction. We repeated the experiments three times and reported averages of the acquired results.

5.1.4. Comparison

We selected following seven state-of-the-art methods to evaluate the effectiveness of SAGE.

[25] Normalized Discounted Cumulative Gain.
[26] Click Through Rate.

- **PER** [9] models HIG and describes user-items connections through meta-path-based feature extraction.
- **CKE** [26] augments matrix factorization with TransR embedding and aggregates collaborative filtering with the concerned knowledge in a combined Bayesian framework.
- **MCRec** [20] co-attentively operates on HIG and exploits meta-paths to acquire the context representations of users and items.
- **RippleNet** [12] is a unified approach of path-based methods and embedding-based methods. It enhances user representations via item information and acquires the required preferences via propagation.
- **KGAT** [15] is an attentive mechanism that acquires initial representations of entities via TransR and performs propagation. It enhances the initial representations of entities via neighborhood information.
- **AKGE** [37] is an attention-based approach and models higher order connections through propagation. It constructs subgraph for experiments based on the concept of "*closer entities are more relevant to each other*".
- **NACF** [38] is a CF-based attentive neighborhood aggregation approach that collects user preferences based on the potential connections among users and items.

5.1.5. Parameterization

We attempted different sets of hyper-parameters with respect to various configurations and finalized optimal combinations of hyper-parameters as summarized in Table 3. We kept $K = [16, 32, 32]$, $d = [32, 64, 64]$, $s = [3, 4, 4]$ and $b = 1024$ for best results on each of the three datasets (i.e., Book-Crossing, Last-FM and Yelp respectively). We identically selected the optimal inputs from the sets of candidate options. For instance, we selected $\varepsilon = 10^{-3}$ from set $\left\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\right\}$, $\lambda = 10^{-7}$ from $\left\{10^{-10}, 10^{-9}, \cdots, 10^{2}, 10^{3}\right\}$, $\eta = 7 \times 10^{-4}$ from $\left\{5 \times 10^{-2}, 5 \times 10^{-3}, 7 \times 10^{-3}, 7 \times 10^{-4}\right\}$, and fixed μ in the range of -0.2 to 0.8 for all datasets. We kept d unchanged for the baseline methods and selected the rest of the hyper-parameters via grid search technique.

5.2. Comparative study (RQ1)

The results achieved through *CTR prediction* (i.e., AUC and Accuracy) are summarized in Table 4 and *top-K recommendation* (i.e., Precision @K, Recall @K and NDCG @K) in Table 5 and Fig. 6. In case of later, extensive experiments are performed on eight different variations of K (i.e., K = 1, 2, 5, 10, 25, 50, 75 and 100), but the results against $K = 5$ and 10 only are represented in Table 5 due to the space limitations and complete analysis of the acquired results is portrayed in Fig. 6.

Relatively, SAGE has outperformed the baseline approaches on all datasets with considerable improvements in all facets of evaluation. With a framework augmented by data purification, propagation, feature learning, retrieval and formalization, SAGE is capable to effectively model higher-order connections among entities and capture the required preferences. SAGE is also capable to access meaningful paths or meta-paths via entity-relevance, retrieve explicit features for semantical synchronization and explicitly convey attentive-weights via the concerned paths or meta-paths. It neither relies on the minimal distance between entities nor accepts the fixed distance weights for similarity estimation like AKGE and NACF respectively. Characteristically, "*a nearer node is more similar*" strategy is adopted to construct the subgraph in AKGE, and NACF calculates inter-entity similarity via the specific weights assigned by the attention mechanism irrespective of the semantic or tendency-based relevancies among entities or their mutual relations. Similarly, KGAT is unable to efficiently attain the hierarchical and sequential structure of HIG due to its recursive propagation. Additionally, it performs extensive aggregation without applying any information relevance check also leads to performance degradation.

NCAF has outperformed AKGE and KGAT in majority of the observations, justifies that the incorporation of data similarity mechanism into the model is better to enhance the performance. However, AKGE and KGAT performed better than RippleNet and MCRec in many cases. It rationalizes that the attention mechanism is healthier in generalizing the underlying model as compared to the straightforward propagation and meta-path-based traversal strategies applied in RippleNet and MCRec respectively.

Similarly, we noticed that RippleNet has outperformed MCRec, CKE and PER in majority of the comparisons. It escalates the dominance of knowledge propagation over the meta-path-based methods and simple regularization-based approaches. And, the enhanced performance of MCRec against CKE and PER dominates meta-path-based selective techniques over simple embedding-based approaches. Lastly, although the general performance of CKE and PER is nearly identical, CKE has a tiny upper edge over PER wrt the empirical avenue.

5.3. Ablation study (RQ2)

In this section, we present the ablation study of the proposed model from three different perspectives.

5.3.1. Impact of individual components

To study the importance of each component of SAGE, we truncated one component at a time and checked the performance of SAGE in the absence of that specific component. Particularly, the obtained results demonstrate that the truncation of *NSP* has significantly degraded the performance of SAGE and depict the importance of information relevance constraint as

Table 3
Adjustment of hyper-parameters for all datasets. The special terms used: sampling size of the influential neighboring (K), data embedding dimension (d), hop-length wrt the path steps (s), KG embedding weight (ε), ENR weight (λ), learning rate (η), batch size (b) & dropout ratio (μ).

Datasets	**Hyper Parameters Setting**
Book-Crossing	$K = 16, d = 32, s = 3, \varepsilon = 10^{-3}, \lambda = 10^{-7}, \eta = 7 \times 10^{-4}, b = 1024, \mu = [-0.2, 08]$
Last-FM	$K = 32, d = 64, s = 4, \varepsilon = 10^{-3}, \lambda = 10^{-7}, \eta = 7 \times 10^{-4}, b = 1024, \mu = [-0.2, 08]$
Yelp	$K = 32, d = 64, s = 4, \varepsilon = 10^{-3}, \lambda = 10^{-7}, \eta = 7 \times 10^{-4}, b = 1024, \mu = [-0.2, 08]$

Table 4
CTR prediction via AUC and Accuracy. The special terms used: Upper-Bound (α), Lower-Bound (β), Mean ($\bar{x}$).

Approaches		**Book-Crossing**		**Last-FM**		**Yelp**	
		AUC	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**
PER		0.6254	0.5921	0.6129	0.5866	0.7254	0.6932
CKE		0.6504	0.6105	0.7389	0.6632	0.7922	0.7098
MCRec		0.7301	0.6619	0.7412	0.6689	0.8011	0.7191
RippleNet		0.7429	0.6691	0.7716	0.6898	0.8213	0.7491
KGAT		0.7203	0.7022	0.8109	0.6791	0.8129	0.7212
AKGE		0.7399	0.7089	0.8015	0.7029	0.8423	0.7547
NACF		*0.7469**	*0.7119**	*0.8201**	*0.7316**	*0.8602**	*0.7749**
SAGE		**0.7697**	**0.7381**	**0.8615**	**0.7643**	**0.8898**	**0.8078**
Improved: (%)-age	α	02.96	03.55	04.81	04.28	03.33	04.07
	β	03.05	03.68	05.05	04.47	03.44	04.25
	$\bar{x}$	**03.01**	**03.61**	**04.93**	**04.37**	**03.38**	**04.16**

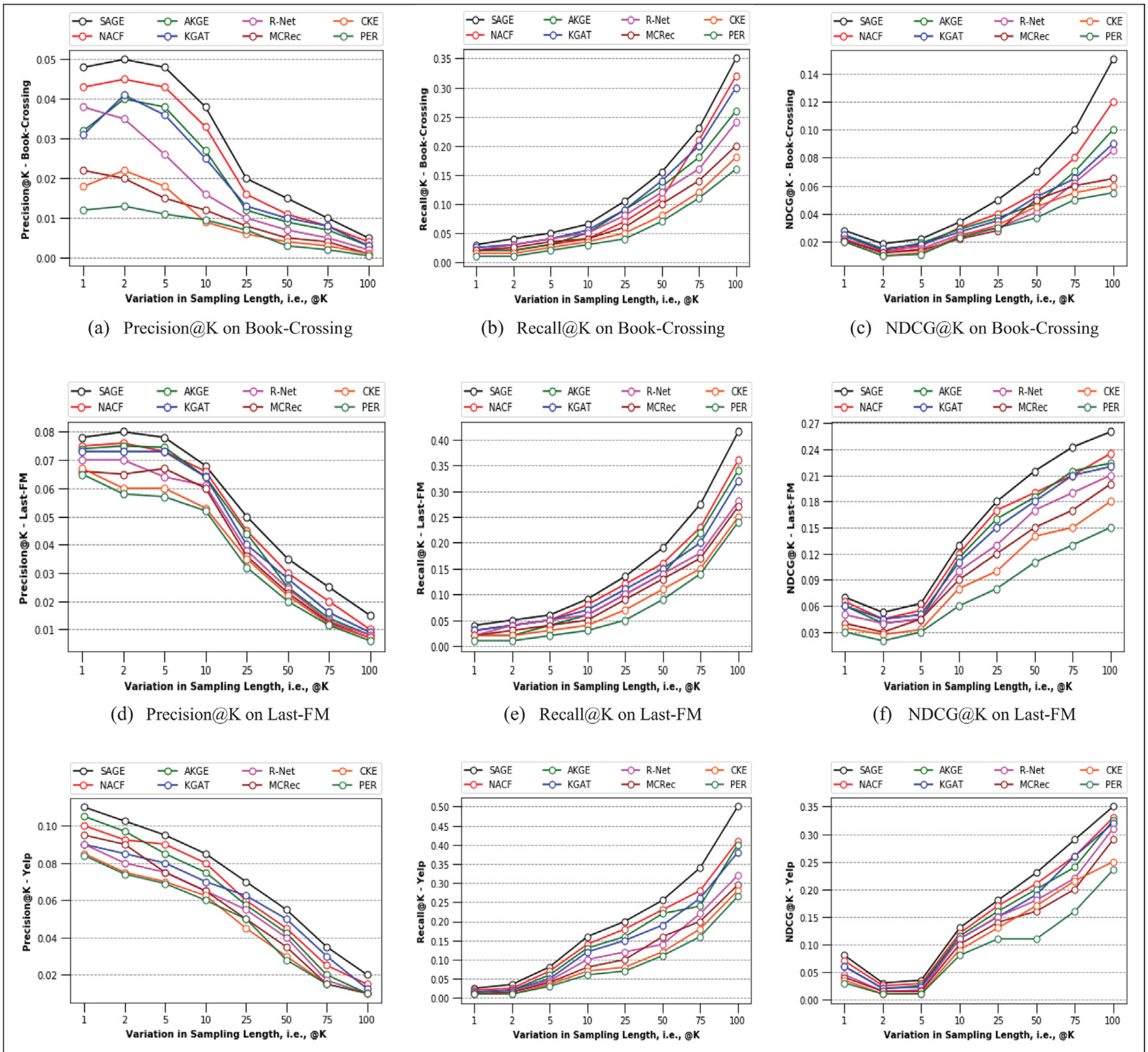
Table 5
Top-K Recommendation via Precision@K, Recall@K and NDCG@K | K = 5 and 10. The special terms used: Upper-Bound (α), Lower-Bound (β), Mean ($\bar{x}$).

Data	**Metrics**	**Approaches**										
		PER	CKE	MCRec	R-Net	KGAT	AKGE	NACF	**SAGE**	Improved: (%)-age		
										α	β	$\bar{x}$
Book-Crossing	Prec@5	0.011	0.018	0.015	0.026	0.036	0.038	*0.043**	**0.048**	10.42	11.63	**11.02**
	Prec@10	0.010	0.009	0.012	0.016	0.025	0.027	*0.033**	**0.038**	13.16	15.15	**14.15**
	Recall@5	0.020	0.025	0.035	0.040	*0.040**	0.030	0.030	**0.046**	13.04	15.00	**14.02**
	Recall@10	0.030	0.035	0.040	0.050	*0.055**	0.050	0.040	**0.063**	12.70	14.55	**13.62**
	NDCG@5	0.011	0.012	0.014	0.015	0.018	*0.019**	0.017	**0.022**	13.64	15.79	**14.71**
	NDCG@10	0.023	0.024	0.022	0.025	0.027	0.029	*0.030**	**0.034**	11.76	13.33	**12.55**
Last-FM	Prec@5	0.057	0.060	0.067	0.064	0.073	*0.075**	0.073	**0.078**	03.85	04.00	**03.92**
	Prec@10	0.052	0.053	0.060	0.061	0.064	0.064	*0.066**	**0.068**	02.94	03.03	**02.99**
	Recall@5	0.020	0.030	0.040	0.050	0.050	0.040	*0.050**	**0.058**	13.79	16.00	**14.90**
	Recall@10	0.030	0.040	0.050	0.060	0.070	0.060	*0.080**	**0.091**	12.09	13.75	**12.92**
	NDCG@5	0.030	0.033	0.045	0.045	0.050	0.045	*0.055**	**0.063**	12.70	14.55	**13.62**
	NDCG@10	0.060	0.080	0.090	0.100	0.110	0.114	*0.119**	**0.130**	08.46	09.24	**08.85**
Yelp	Prec@5	0.069	0.070	0.075	0.075	0.080	0.085	*0.090**	**0.095**	05.26	05.56	**05.41**
	Prec@10	0.060	0.063	0.065	0.065	0.070	0.075	*0.080**	**0.085**	05.88	06.25	**06.07**
	Recall@5	0.030	0.035	0.040	0.045	0.050	0.060	*0.070**	**0.081**	13.58	15.71	**14.65**
	Recall@10	0.060	0.070	0.080	0.100	0.120	0.130	*0.140**	**0.160**	12.50	14.29	**13.39**
	NDCG@5	0.010	0.011	0.015	0.017	0.023	0.025	*0.029**	**0.033**	12.12	13.79	**12.96**
	NDCG@10	0.080	0.090	0.100	0.110	0.113	0.115	*0.120**	**0.130**	07.69	08.33	**08.01**

shown in Table 6. Similarly, the truncation of *FIE* and Gated Formalizer (*GF*) also disclosed their significance to the enhancement of performance. Moreover, the truncation of Influential-nodes' Sampler (IS) and the replacement of Swish Activator (*SA*) also slightly decreased the performance.

5.3.2. Impact of prominent path acquisition

We compared the proposed NSP modeling technique with four different path modeling methods via SAGE as its variants, i.e., $\mathrm{SAGE}_{\mathrm{MPR}}$, $\mathrm{SAGE}_{\mathrm{MPS}}$, $\mathrm{SAGE}_{\mathrm{MPRW}}$ and $\mathrm{SAGE}_{\mathrm{SDES}}$ and reported the performance comparison in Table 7. First, we used Meta Path-based Resemblance (MPR) to construct local subgraph with relevant meta-paths based on the mutual similarity between them as proposed by SemRec [13]. Second, we utilized Meta Path-based Selection (MPS) to create subgraph from the selected meta-paths as introduced by FMG [23]. Third, we exploited Mata Path-based Random Walk (MPRW) strategy to retrieve salient paths from KG and construct the subgraph according to HERec [27]. Fourth, we applied Shortest Distance-



(a) Precision@K on Book-Crossing (b) Recall@K on Book-Crossing (c) NDCG@K on Book-Crossing

(d) Precision@K on Last-FM (e) Recall@K on Last-FM (f) NDCG@K on Last-FM

(g) Precision@K on Yelp (h) Recall@K on Yelp (i) NDCG@K on Yelp

Fig. 6. Results-Analysis of top-K recommendation via Precision@K, Recall@K and NDCG@K | K = [1,2,5,10,25,50,75, and, 100] on Book-Crossing, Last-FM and Yelp datasets.

Table 6

Performance comparison among different variants of SAGE. Note: The truncation of a component is symbolized as the subtraction of that component from SAGE (e.g., SAGE $_{-\ \mathrm{NSP}}$ means SAGE without NSP and so on).

SAGE Variants	Book-Crossing		Last-FM		Yelp	
	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy
SAGE $_{-\ \mathrm{NSP}}$	0.7281	0.7011	0.8232	0.7319	0.8517	0.7721
SAGE $_{-\ \mathrm{FIE}}$	0.7582	0.7262	0.8499	0.7531	0.8782	0.7923
SAGE $_{-\ \mathrm{GF}}$	0.7602	0.7309	0.8535	0.7565	0.8824	0.7998
SAGE $_{-\ \mathrm{IS}}$	0.7612	0.7319	0.8542	0.7573	0.8837	0.8019
SAGE $_{-\ \mathrm{SA}}$	0.7628	0.7323	0.8565	0.7592	0.8849	0.8032
SAGE	**0.7697**	**0.7381**	**0.8615**	**0.7643**	**0.8898**	**0.8078**

based Entity Selection (SDES) technique to acquire prominent paths from the data and use them to construct the subgraph as applied by AKGE [37]. The experimental results show that there is a nominal and onward increasing difference in the performance of $\mathrm{SAGE_{MPR}}$, $\mathrm{SAGE_{MPS}}$ and $\mathrm{SAGE_{MPRW}}$. But, $\mathrm{SAGE_{SDES}}$ achieved a bit better results showing the dominance of shortest

Table 7
Performance comparison among different variants of SAGE wrt the path selection strategies.

SAGE Variants	Book-Crossing		Last-FM		Yelp	
	AUC	Accuracy	AUC	Accuracy	AUC	Accuracy
$SAGE_{MPR}$	0.7074	0.6695	0.7599	0.6637	0.7959	0.7110
$SAGE_{MPS}$	0.7153	0.6756	0.7645	0.6701	0.8067	0.7187
$SAGE_{MPRW}$	0.7268	0.6819	0.7735	0.6792	0.8189	0.7299
$SAGE_{SDES}$	0.7409	0.7099	0.8085	0.7084	0.8492	0.7602
SAGE	**0.7697**	**0.7381**	**0.8615**	**0.7643**	**0.8898**	**0.8078**

distance-based node-selection over the previous three methods wrt the performance. Nevertheless, the proposed approach (i.e., $SAGE_{NSP}$) has outperformed the above mentioned variants of SAGE and highlighted the significance of similarity-guided data collection.

5.3.3. Impact of SRF vs traditional aggregators

To highlight the effectiveness of SRF against the traditional aggregators, we truncated SRF and individually incorporated each traditional aggregator into SAGE as $SAGE_N$, $SAGE_C$, $SAGE_S$ and $SAGE_{SP}$ and reported the respective performance in Table 8. It is evident from the experimental results that there is a minimal and continuous increasing difference in the performance from $SAGE_N$ to $SAGE_{SP}$ on all datasets. Similarly, we applied SAGE via SRF (i.e., SAGE) and also reported the achieved results in Table 8. It is worth mentioning that SRF is described in Section 4.5. The results' analysis demonstrates that SRF has outperformed the traditional aggregators by highlighting the positive influence of SRF on recommendation performance.

5.4. Sensitivity study (RQ3)

We describe how different adjustments of hyper-parameters effect the performance of SAGE.

5.4.1. Sampling size of influential neighboring

To discover the impact of sampling size of influential neighboring on the general performance of SAGE, we represented the sampling length through K individual units as $K = 2^i | i = \{1, 2, 3, 4, 5, 6, 7\}$ and performed the experiments. Empirically, we observed a rapid increase in the performance with increase in K up to $i = 5$ as shown in Table 9. On $i = 5$, SAGE acquired the highest performance and from $i = 4$ to 6 including $i = 5$, it preserved an optimal and higher performance. However, the general performance is slightly decreased after $i = 5$ and highly declined after $i = 6$. Therefore, it can be concluded that on $i > 6$, the model is unable to feasibly generalize the information structure due to losing its grip on HIG.

5.4.2. Embedding depth of underlying information

We investigated how different depths of information embedding influence the performance of SAGE. We embedded the datasets to d levels as $d = 2^i | i = \{1, 2, 3, 4, 5, 6, 7\}$, performed the experiments wrt the embedding depths and analyzed the achieved results as shown in Table 10. Empirically, we observed that the performance is continuously increasing with each increase in i up to $i = 5$. At $i > 5$, we noticed a slight down fall wrt good number observations up to $i = 6$ in the performance, but still it remained in a persisting condition. However, we observed a remarkable degradation in the performance wrt majority of observations on $i > 6$. It implies that the model has lost its grip on information structure after $i = 6$ and collected the noise consequently. Therefore, we conclude that the proposed model can effectively generalize the information up to $i = 6$.

5.4.3. Path depth of higher-order connectivity

We evaluated the performance of the proposed approach on different depths of higher-order connectivity and summarized the comparison in Table 11. The path depth $s | s = 1, 2, \cdots, 5$ represents s onward hops through a specified meta-path starting from the n_c. Initially, we observed that the performance of SAGE is rapidly increasing with each increase in s up to $s = 3$ according to all of the achieved outcomes. Although the performance is slightly decreased on $s = 4$ for many instances, SAGE interchangeably obtained the highest performance on $s = 3$ and 4. However, the model faced a huge down fall after $s = 4$ and obtained the worst performance on $s = 5$. It implies that the model is unable to effectively capture the hierarchical structure of HIG and consequently suffers from over-fitting on $s > 4$.

5.5. Robustness study (RQ4)

In this section, we study the performance of SAGE against cold-start and data-sparsity problems.

Table 8
Performance comparison among different variants of SAGE wrt aggregation. Note: $SAGE_N$, $SAGE_C$, $SAGE_S$ and $SAGE_{SP}$ represent SAGE via Neighbor (*N*), Concatenation (*C*), Sum (*S*) and Sum&Product (*SP*) aggregators respectively.

Impact of Aggregators	**Book-Crossing**		**Last-FM**		**Yelp**	
	AUC	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**
$SAGE_N$	0.7501	0.7188	0.8407	0.7414	0.8701	0.7818
$SAGE_C$	0.7519	0.7209	0.8422	0.7428	0.8717	0.7857
$SAGE_S$	0.7533	0.7223	0.8462	0.7442	0.8723	0.7917
$SAGE_{SP}$	0.7541	0.7236	0.8499	0.7499	0.8774	0.7952
SAGE	**0.7697**	**0.7381**	**0.8615**	**0.7643**	**0.8898**	**0.8078**

Table 9
SAGE's performance wrt the sampling size K of Influential Neighboring.

K	**Book-Crossing**		**Last-FM**		**Yelp**	
	AUC	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**
2	0.7605	0.7287	0.8503	0.7556	0.8794	0.7963
4	0.7629	0.7307	0.8545	0.7584	0.8827	0.8007
8	0.7657	0.7335	0.8578	0.7614	0.8864	0.8032
16	0.7692	**0.7381**	0.8611	0.7641	0.8892	**0.8078**
32	**0.7697**	0.7378	**0.8615**	0.7632	**0.8898**	0.8077
64	0.7687	0.7375	0.8609	**0.7643**	0.8887	0.8064
128	0.7354	0.7009	0.8372	0.7298	0.8496	0.7754

Table 10
SAGE's performance wrt the embedding size of the underlying data.

d	**Book-Crossing**		**Last-FM**		**Yelp**	
	AUC	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**
2	0.7634	0.7321	0.8498	0.7432	0.8817	0.7992
4	0.7671	0.7352	0.8532	0.7564	0.8862	0.8019
8	0.7665	0.7342	0.8577	0.7612	0.8857	0.8034
16	**0.7697**	**0.7381**	0.8604	0.7637	0.8874	0.8059
32	0.7692	0.7352	**0.8615**	0.7641	**0.8898**	**0.8078**
64	0.7630	0.7303	0.8543	**0.7643**	0.8872	0.8029
128	0.7453	0.7198	0.8395	0.7394	0.8643	0.7793

Table 11
SAGE's performance wrt the Path Step ***s*** of the meta-Paths.

s	**Book-Crossing**		**Last-FM**		**Yelp**	
	AUC	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**
1	0.7342	0.7212	0.8456	0.7376	0.8683	0.7832
2	0.7533	0.7309	0.8599	0.7537	0.8812	0.7998
3	0.7689	**0.7381**	**0.8615**	0.7621	**0.8898**	**0.8078**
4	**0.7697**	0.7345	0.8594	**0.7643**	0.8852	0.8034
5	0.6838	0.6653	0.7432	0.6739	0.7942	0.7152

5.5.1. Robustness against cold start problem

We performed online experiments on ACRD using the applied models and summarized the results' analysis in Table 12 and presented a detailed description of the achieved results via online evaluation metrics in Fig. 7. We applied CTR prediction via AUC, Accuracy, IAGRQ and IASRU metrics to online evaluate the performance of SAGE and the baseline methods. Concerning the cold-start problem, we used IAGRQ and IASRU to appraise the applied models at "*different time intervals of a given timespan*" and "*the entire timespan*" respectively in a specific recommendation scenario. The smaller time intervals are less prone to the cold start problem but suffer from trade-off between the amount of new information and the model's generalization capability. To concurrently maintain the updated information and the model's generalization capability, it is compulsory to sequentially keep the track of individual time intervals. Typically, the performance of IASRU depends on the results of IAGRQ since the individual time intervals of each timespan of IAGRQ continuously refreshes the entire timespan of IASRU. It is worth notifying that one recommendation scenario spreads over one timespan and one timespan is equal to 150 min. The experimental results demonstrate that the proposed approach has outperformed the baseline methods with satisfactory

Table 12
Performance of the applied models on ACRD via CTR Prediction. The special terms used: Upper-Bound (α), Lower-Bound (β), Mean ($\bar{x}$).

Approaches		AUC	Accuracy	IAGRQ%	IASRU%
MCRec		0.6978	0.6325	1.1852	1.9825
RippleNet		0.7118	0.6498	1.6898	2.8263
KGAT		0.7072	0.6710	2.0785	3.2562
AKGE		0.7028	0.6797	2.6639	3.6897
NACF		*0.7142**	*0.6872**	*3.2756**	*4.0649**
SAGE		**0.7359**	**0.7019**	**3.5629**	**4.4286**
Improved: (%)-age	α	2.9488	2.0943	8.0637	8.2125
	β	3.0384	2.1391	8.7709	8.9473
	$\bar{x}$	**2.9936**	**2.1167**	**8.4173**	**8.5799**

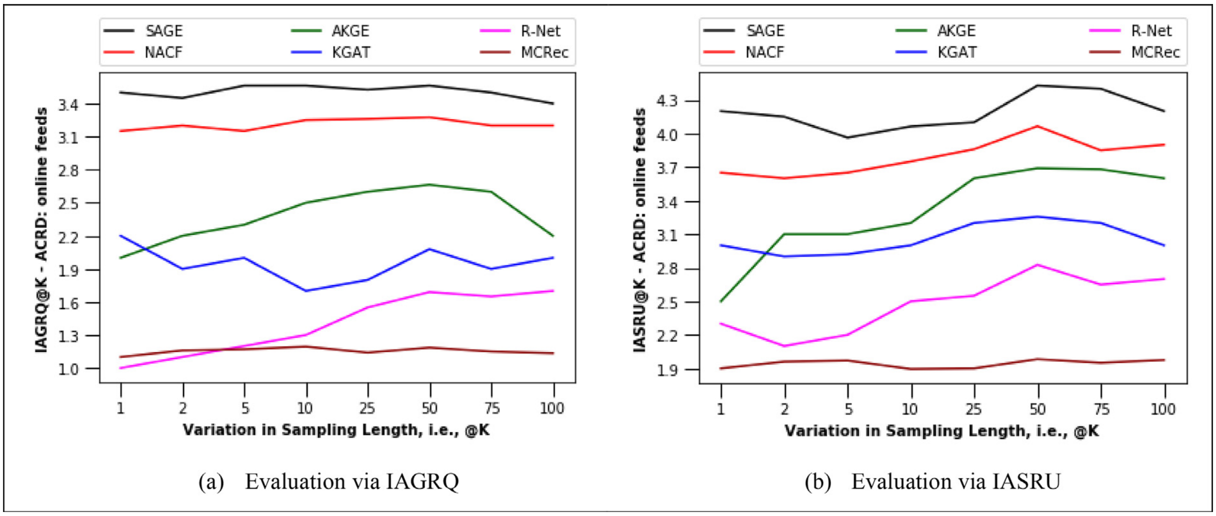


Fig. 7. Online performance evaluation of the proposed approach and the baseline models through IAGRQ and IASRU at *"different intervals of a given timespan"* and *"the entire timespan"* respectively. A single timespan is equal to 150 min.

improvements on ACRD in online evaluation. Therefore, it implies that SAGE is highly effective against cold-start problem in online recommendation scenarios.

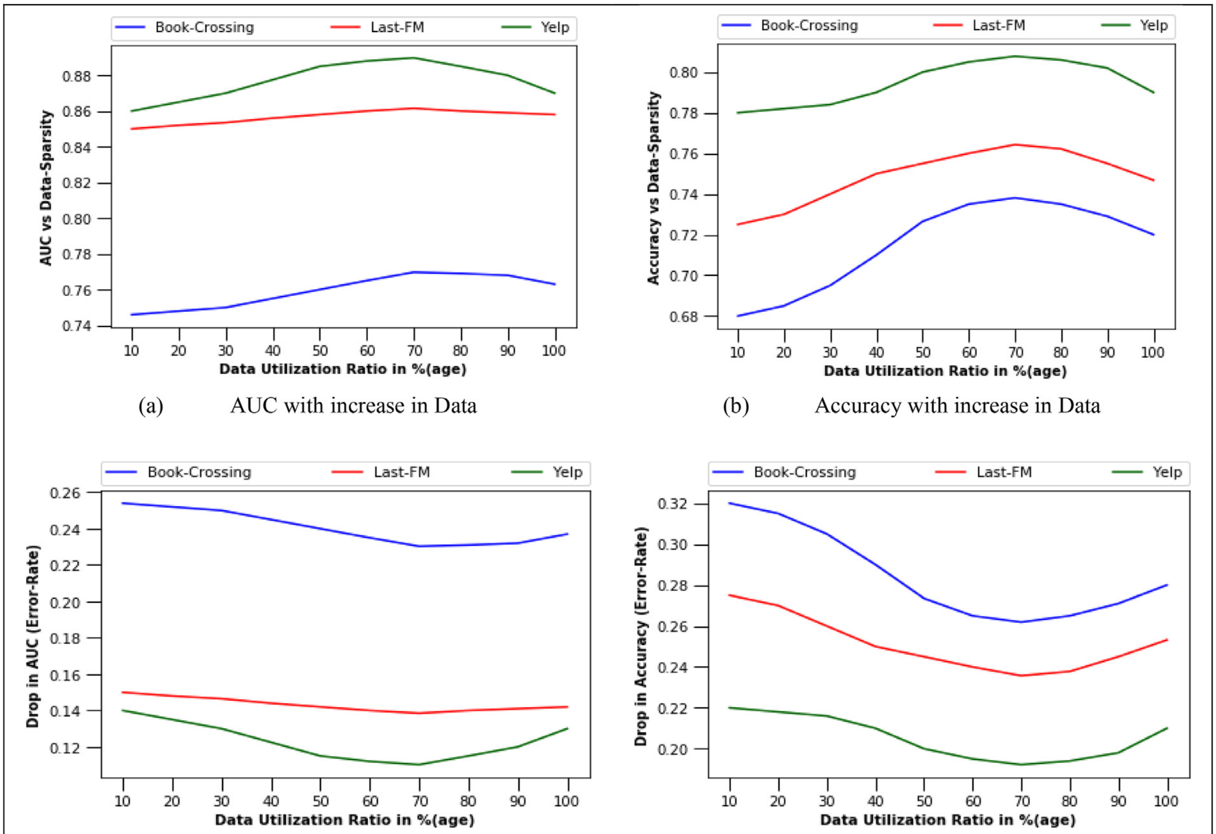
5.5.2. Robustness against data sparsity problem

The structural mechanism and underlying modules of SAGE (especially information relevance constraint application and IKN) are vigorous against the problem of data sparsity. Different experimental results (as shown in Tables 4 and 5, and Figs. 6 and 8, etc.) demonstrate that the performance of SAGE is sufficient against data sparsity on the used datasets. For instance, as shown in Fig. 8(a) and 8(b), the performance (i.e., the results achieved via AUC and Accuracy) has uninterruptedly increased on all datasets up to a 70 % share of the Data Utilization (DU) where the model achieved its highest performance. Exactly after the 70 % DU and onward, the general performance of SAGE has gradually decreased on about all of the applied datasets. As a result, it implies that while handling a larger size of the data, SAGE has lost its grip on the semantic and hierarchical structure of information that triggered the overfitting. On the other hand, the error rate wrt AUC and Accuracy is deemed as the additive inverse of the results of AUC and Accuracy respectively as shown in Fig. 8(c) and 8(d). Conclusively, we are eager to assert that the proposed approach is capable of effectively dealing with the data sparsity problem.

5.6. Case study (RQ5)

The proposed approach is capable of providing reasonable explanations about the process it utilizes to deliver recommendations. Therefore, we demonstrate a recommendation scenario in Fig. 10 based on the toy example presented in Fig. 1 to clarify the explainability phenomenon of the SAGE. Typically, we retrieved the following five random click-samples (i.e., the targeted items) from users-item interaction log:

c1: "OOP plays a central role in data structures and algorithms".
c2: "O'Reilly, also publishes books, possesses equal importance in computer science as well"
c3: "Robert Lafore has a remarkable contribution to the programming languages".
c4: "Java byte-code provides platform portability".



(a) AUC with increase in Data (b) Accuracy with increase in Data

(c) Error Rate in AUC with increase in Data (d) Error Rate in Accuracy with increase in Data

Fig. 8. Performance of SAGE on Book-Crossing, Last-FM and Yelp Datasets against the Data-Sparsity Problem.

c5: "Java Script is interpreter-based language supports OOP".

The boldface-words in each click-sample conform to the entity-names in KG and the information interconnecting them carries resemblance to the relations among entities. The space-less strings of the boldface-words and their interconnecting information is embedded as a subgraph to the vector space and mapped its low level representations to the relevant information in the external knowledge repository (i.e., MS Satori) to enrich it. A raw sketch of the exploitation of paths and meta-paths in recommendation generation is represented in Fig. 9 that demonstrates three different forms of the path reasoning, i.e., (a) single-hop inferring paths, (b) intermediate meta-paths, and (c) recommendation paths that connect start nodes (users) with end nodes (books). To infer the potential preferences, interactional sequences are used as the reasoning facts between start and end nodes of the meta-paths (i.e., from interacting users to recommended books).

Particularly, the concerned reasoning meta-paths are mentioned in Fig. 9 that provide sequential inferences from users to items and suggest potential preferences about the users' interests. Therefore, paths that possibly possessed the highest relevance weights among nodes are distinguishingly highlighted in red and the rest of the paths are removed from the KG as shown in Fig. 10 (a). Once the recommendations are finalized, the intermediate nodes between start and end nodes of the concerned meta-paths are truncated and single-hop connections between start and end nodes are attained to portray straight recommendations between the given nodes as shown in Fig. 10(b). For instance, in Fig. 10(a), a meta-path "*Ahmad* → *ALP for IBM* → *Robert Lafore* → *Programming* → *OOP with Java*" infers that "*OOP with Java*" should be recommended to "*Ahmad*". Therefore, the intermediate nodes "*ALP for IBM*", "*Robert Lafore*" and "*Programming*" between "*Ahmad*" and "*OOP with Java*" are truncated and "*Ahmad* → *OOP with Java*" is attained as the recommendation path implies that "*OOP with Java*" is recommended to "*Ahmad*" as shown in Fig. 10(b). The formal graphical representation of reasoning meta-paths and recommendation paths is portrayed in Fig. 10(a) and 10(b) respectively. Hence, based on the described case study, it can be concluded that the proposed approach is capable of providing satisfactory explanations about its delivered recommendations.

6. Conclusion and Future-work

To tackle information relevance and semantical structure of KG at n-order connections, we proposed SAGE – a novel information relevance enhanced KGEE approach for recommendation – to effectively model HIG in an end-to-end automatic way.

```
<"Asad":"H/F Java":"Publisher":"H/F Java-Script"><"Asad":"H/F Java":"Publisher":"H/F
Java-Script":"Programming":"OOP with Java"><"Asad":"H/F Java":"Programming":"OOP with
Java"><"Ahmad":"Asad":"H/F Java":"Publisher":"H/F Java-Script"><"Ahmad":"Asad":"H/F
Java":"Publisher":"H/F Java-Script":"Programming":"OOP with Java"><"Ahmad":"Asad":"H/F
Java":"Programming":"OOP with Java"><"Ahmad":"OOP in C++":"Programming":"OOP with Java
"><"Ahmad":"OOP in C++":"Robert Lafore":"Programming":"OOP with Java"><"Ahmad":"ALP for
IBM":"Robert Lafore":"Programming":"OOP with Java"><"Ahmad":"ALP for IBM":"Robert Lafore
":"DS & Algo"><"Ahmad":"OOP in C++":"Robert Lafore":"DS & Algo">

<"Asad":"H/F Java-Script"><"Asad":"OOP with Java"><"Asad":"OOP with Java"><"Ahmad":"H/F
Java-Script"><"Ahmad":"OOP with Java"><"Ahmad":"OOP with Java"><"Ahmad":"OOP withJava"><"
Ahmad":"OOP with Java"><"Ahmad":"OOP with Java"><"Ahmad":"DS & Algo"><"Ahmad":"DS & Algo">

<"Asad":"H/F Java-Script"><"Asad":"OOP with Java"><"Ahmad":"H/F Java-Script"><"Ahmad":"OOP
with Java"><"Ahmad":"DS & Algo">
```

Fig. 9. Raw demonstration of the possible meta-paths between start and end nodes. Particularly, in the upper part, all possible meta-paths are summarized; in the middle part, all of the single-hop paths between the start and end nodes are demonstrated; and in the lower part, only the meta-paths with the highest weights are represented as the recommendation paths between users and items.

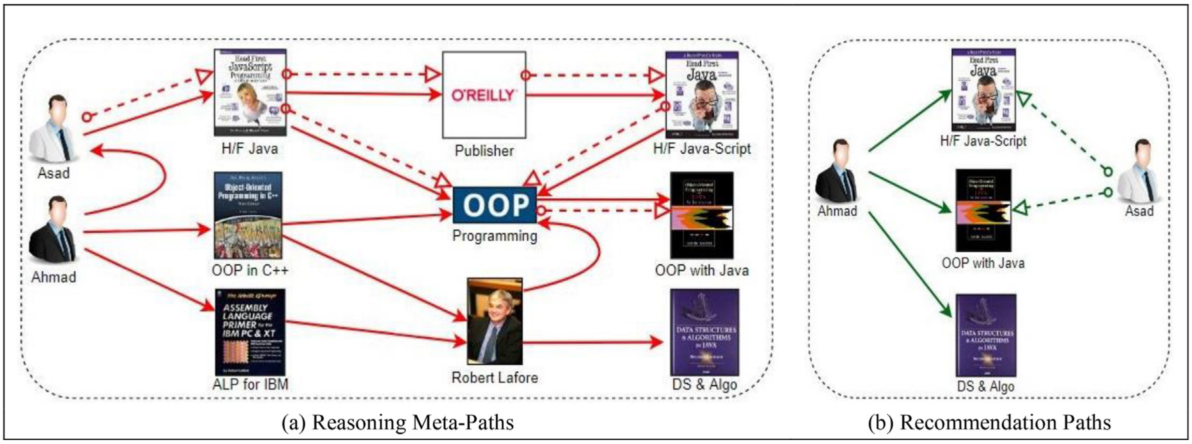


Fig. 10. Graphical representation of reasoning meta-paths and recommendation paths of the given case study. For clarification, we used different path styles for different users in reasoning as well as in recommendation (e.g., solid lines for Ahmad and dashed lines for Asad).

The proposed approach contains four modules, i.e., *Similarity-guided Subgraph Construction* (SSC), *Interaction-enhanced Knowledge Network* (IKN), Explicit *Feedback-attributed Interaction Enhancement* (FIE) technique, and *Sampling, Retrieval and Formalization* (SRF) suite. In SSC, we constructed a local subgraph for experiments, based on node-to-node and node-to-path semantic and structural similarity. For this purpose, we proposed *NSP-modeling* technique to automatically find higher-order relevant nodes in the provided data and fold them into the unique meta-paths. The unique meta-paths are combined into the subgraph. IKN propagated interaction-enhanced user-item information over the subgraph to discover potential preferences of users about unseen items of their interests. FIE exploited explicit semantic features (i.e., user feedbacks/ratings extracted from KGE) to concurrently enhance IKN and model's regularization capability to highlight influential nodes via GNN. SRF is used to sample the influential nodes in IKN, independently move their preferences to LCN, and streamline all LCN to the main unit via gated formalization technique. Finally, the main unit is used to predict the consequent preferences for recommendations. We performed extensive experiments on real-word benchmark datasets to evaluate the effectiveness of SAGE in comparison with the state-of-the-art methodologies. The experimental results demonstrate that the proposed approach has outperformed the baseline methods with remarkable improvements by effectively meeting the highlighted challenges.

In future work, we plan to (a) reduce time complexity of SAGE via binary computational (hashing) approach, (b) exception handle SAGE to work on general databases besides the pre-processed (synthesized) datasets, and (c) more precisely extract HIG data to preserve structural and semantic granularity of the triplets, up to a maximum possible extent, via information encapsulation and deep-CF techniques. Moreover, to rid subgraph construction, we aim to focus on information relevance-enhanced KG-completion-cum-recommendation approach.

CRediT authorship contribution statement

Nasrullah Khan: Conceptualization, Methodology, Writing – original draft. **Zongmin Ma:** Supervision, Validation, Writing – review & editing. **Aman Ullah:** Validation. **Kemal Polat:** Writing – review & editing.

Data availability

Data link is provided in the Section of "Benchmark datasets."

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The research work was supported in part by the Basic Research Program of Jiangsu Province (BK20191274) and the National Natural Science Foundation of China (62176121 and 61772269).

References

[1] L. Sang, M. Xu, S. Qian, X. Wu, Knowledge graph enhanced neural collaborative recommendation, Expert Syst. Appl. 164 (2021) 113992.
[2] J. Chen, J. Yu, W. Lu, Y. Qian, P. Li, Ir-rec: An interpretive rules-guided recommendation over knowledge graph, Inf. Sci. 563 (2021) 326–341.
[3] C. Wu, S. Liu, Z. Zeng, M. Chen, A. Alhudhaif, X. Tang, F. Alenezi, N. Alnaim, X. Peng, Knowledge graph-based multi-context-aware recommendation algorithm, Inf. Sci. 595 (2022) 179–194.
[4] X. Zheng, M. Wang, C. Chen, Y. Wang, Z. Cheng, Explore: Explainable item-tag co-recommendation, Inf. Sci. 474 (2019) 170–186.
[5] H. Wang, H. Jhou, Y. Tsai, Adapting topic map and social influence to the personalized hybrid recommender system, Inf. Sci. 575 (2021) 762–778.
[6] S. Wang, M. Gong, Y. Wu, M. Zhang, Multi-objective optimization for location-based and preferences-aware recommendation, Inf. Sci. 513 (2020) 614–626.
[7] W. Yuan, H. Wang, X. Yu, N. Liu, Z. Li, Attention-based context-aware sequential recommendation model, Inf. Sci. 510 (2020) 122–134.
[8] R. Dridi, S. Zammali, T. Alsulimani, K. Arour, Effective rating prediction based on selective contextual information, Inf. Sci. 510 (2020) 218–242.
[9] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, J. Han, Personalized entity recommendation: A heterogeneous information network approach, in: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, 2014, pp. 283–292.
[10] N. Khan, Z. Ma, A. Ullah, K. Polat, DCA-IoMT: Knowledge Graph Embedding-enhanced Deep Collaborative Alerts-recommendation against COVID19, IEEE Trans. Ind. Inf. (2022).
[11] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, T.-S. Chua, Explainable reasoning over knowledge graphs for recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, 2019, pp. 5329-5336.
[12] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, M. Guo, Exploring high-order user preference on the knowledge graph for recommender systems, ACM Trans. Info. Systems (TOIS) 37 (3) (2019) 1–26.
[13] C. Shi, Z. Zhang, P. Luo, P. Yu, Y. Yue, B. Wu, Semantic path based personalized recommendation on weighted heterogeneous information networks, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015, pp. 453–462.
[14] J. Huang, W.X. Zhao, H. Dou, J.-R. Wen, E.Y. Chang, Improving sequential recommendation with knowledge-enhanced memory networks, in: The 41st international ACM SIGIR Conference on Research & Development in Information Retrieval, 2018, pp. 505–514.
[15] X. Wang, X. He, Y. Cao, M. Liu, T.-S. Chua, Kgat: Knowledge graph attention network for recommendation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019, pp. 950–958.
[16] X. Guo, W. Lin, Y. Li, Z. Liu, L. Yang, S. Zhao, Z. Zhu, Dken: Deep knowledge-enhanced network for recommender systems, Inf. Sci. 540 (2020) 263–277.
[17] N. Khan, Z. Ma, L. Yan, A. Ullah, Hashing-based semantic relevance attributed knowledge graph embedding enhancement for deep probabilistic recommendation, Appl. Intelligence (2022) 1–26.
[18] J. Zhang, M. Gao, J. Yu, L. Yang, Z. Wang, Q. Xiong, Path-based reasoning over heterogeneous networks for recommendation via bidirectional modeling, Neurocomputing 461 (2021) 438–449.
[19] X. Huang, S. Qian, Q. Fang, J. Sang, C. Xu, Meta-path augmented sequential recommendation with contextual co-attention network, ACM Trans. Multimedia Comp., Commun., Appl. (TOMM) 16 (2) (2020) 1–24.
[20] B. Hu, C. Shi, W. Zhao, P. Yu, Leveraging meta-path based context for top-N recommendation with a neural co-attention model, in: Proceedings of 24nd ACM Int. Conference on Knowledge Discovery and Data Mining, 2018, pp. 1531–1540.
[21] J. Han, L. Zheng, H. Huang, Y. Xu, P. Yu, W. Zuo, Deep latent factor model with hierarchical similarity measure for recommender systems, Inf. Sci. 503 (2019) 521–532.
[22] K. Kang, J. Park, W. Kim, H. Choe, J. Choo, Recommender system using sequential and global preference via attention mechanism and topic modeling, Proceedings of 28th ACM International Conference, 2019, pp. 1–10.
[23] H. Zhao, Q. Yao, J. Li, Y. Song, D.L. Lee, Meta-graph based recommendation fusion over heterogeneous information networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 635–644.
[24] S. Yang, W. Zhang, R. Tang, M. Zhang, Z. Huang, Approximate inferring with confidence predicting based on uncertain knowledge graph embedding, Inf. Sci. 609 (2022) 679–690.
[25] N. Khan, Z. Ma, A. Ullah, K. Polat, Categorization of knowledge graph based recommendation methods and benchmark datasets from the perspectives of application scenarios: a comprehensive survey, Expert Syst. Appl. 206 (2022) 117737.
[26] F. Zhang, N.J. Yuan, D. Lian, X. Xie, W.-Y. Ma, Collaborative knowledge base embedding for recommender systems, in: Proceedings of 22nd ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining, 2016, pp. 353–362.
[27] C. Shi, B. Hu, W.X. Zhao, P.S. Yu, Heterogeneous information network embedding for recommendation, IEEE Trans. Knowl. Data Eng. 31 (2) (2019) 357–370.
[28] T. Fu, W.-C. Lee, Z. Lei, Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning, in: Proceedings of the 2017 ACM on Conf. on Information and Knowledge Management, 2017, pp. 1797–1806.
[29] X. Xin, X. He, Y. Zhang, Y. Zhang, J. Jose, Relational collaborative filtering: Modeling multiple item relations for recommendation, in: Proceedings of the 42nd international ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 125–134.

[30] Y. Cao, X. Wang, X. He, Z. Hu, T.-S. Chua, Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences, in: The World Wide Web conference, 2019, pp. 151–161.
[31] J.Y. Zhu Sun, J. Zhang, A. Bozzon, L.-K. Huang, C. Xu, Recurrent knowledge graph embedding for effective recommendation, in: ProceedIngs of the 12th ACM Conference on Recommender Systems, 2018, pp. 297–305.
[32] H. Wang, F. Zhang, X. Xie, M. Guo, Dkn: Deep knowledge-aware network for news recommendation, in: Proceedings of the 2018 Conference on World Wide Web, 2018, pp. 1835–1844.
[33] L. Chen, L. Wu, R. Hong, K. Zhang, M. Wang, Revisiting Graph Based Collaborative Filtering: A Linear Residual Graph Convolutional Network Approach, The Thirty-Fourth AAAI Conference on Artificial Intelligence, 2020, Vol. 34, No. 01, pp. 27–34.
[34] H. Wang, M. Zhao, X. Xie, W. Li, M. Guo, Knowledge graph convolutional networks for recommender systems, in: The World Wide Web conference, 2019, pp. 3307–3313.
[35] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, Z. Wang, Knowledge-aware graph neural networks with label smoothness regularization for recommender systems, in: Proceedings of 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 968–977.
[36] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: Proceedings of the 43rd international ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 639–648.
[37] X. Sha, Z. Sun, J. Zhang, Attentive knowledge graph embedding for personalized recommendation, arXiv preprint: 1910.08288 (2019).
[38] D. Zhang, L. Liu, Q. Wei, Y. Yang, P. Yang, Q. Liu, Neighborhood aggregation collaborative filtering based on knowledge graph, Appl. Sci. 10 (2020) 3818.
[39] P. Ramachandran, B. Zoph, Q. V. Le, Searching for activation functions, arXiv:1710.05941 (2017).
[40] W.X. Zhao, G. He, K. Yang, H. Dou, J. Huang, S. Ouyang, J.-R. Wen, Kb4rec: A data set for linking knowledge bases with recommender systems, Data Intelligence 1 (2019) 121–136.