ScholarFinder: Knowledge Embedding based Recommendations using a Deep Embedded Clustering Model

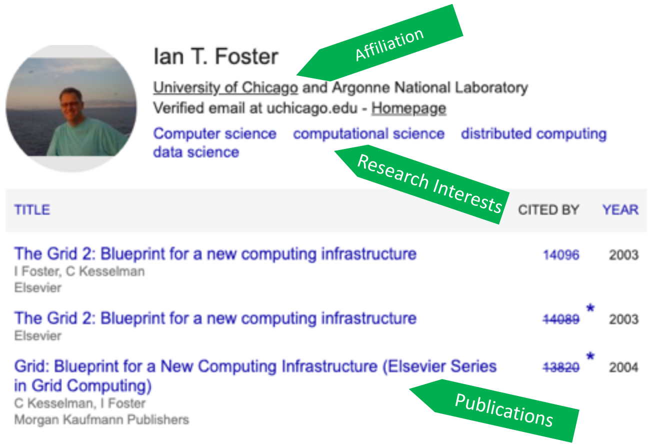
Yuanxun Zhang, Xiyao Cheng, Roland Oruche, Sai Swathi Sivarathri, Prasad Calyam

***Abstract*—Bold scientific research tasks need multi-disciplinary knowledge and collaborations that require finding scholars from particular domains with relevant knowledge. Given the variety of scholars and diversity, finding the appropriate scholar is an important and challenging problem for scientific communities. In this paper, we propose a "ScholarFinder" framework that uses contextual information (abstracts or publications) for embedding a scholar's knowledge in an unsupervised learning manner. Specifically, we implement an unsupervised embedding technique viz., *Variational AutoEncoder (VAE)*. For better feature representation learning, we also implement a *Variational Deep Embedded Clustering (VDEC)* method that further enhances downstream tasks (e.g., clustering, classification) accuracy, scalability, and performance. In addition, we incorporate a multi-task learning scheme into our VDEC model for improving the effectiveness of simultaneously learning both embedding and clustering. Subsequently, the downstream tasks can be built based on pre-trained scholars' knowledge embeddings to predict suitability of a scholar for a research task. Using a dataset involving a 20-year collection of federal grant awards, we have demonstrated how our pre-trained model improved the performance for downstream tasks. We have also investigated how our pre-trained model can be integrated into a knowledge graph to achieve better performance. Lastly, we show that our ScholarFinder model variants outperform state-of-the-art baseline models (i.e., XGBoost, GBDT, AdaBoost, DNN, GraphSAGE, DEC, VaDE) and recent LLM based models (i.e., Bert4Rec, OpenP5) by atleast 18%.**

***Index Terms*—Embedding, Deep Learning, Representation Learning, Recommendation System, Clustering.**

I. Introduction

KNOWLEDGE creation becomes possible due to access to experts who can answer relevant research questions. Moreover, knowledge creation today involves experts working on bold problems that require interdisciplinary expertise and cross-domain collaborations. However, finding relevant scholars across scientific domains to execute critical research tasks is demanding. For example, how do we find pertinent experts to effectively work on building computational bioinformatics/neuroscience infrastructure for flexibly scaling data analysis pipelines? Answering such a scholar finder problem is a hard challenge for handling diverse and interdisciplinary scientific research tasks, especially when identifying relevant experts from multiple domains in a pool of several thousand scholars.



Ian T. Foster
University of Chicago and Argonne National Laboratory
Verified email at uchicago.edu - Homepage
Computer science computational science distributed computing data science

TITLE	CITED BY	YEAR
The Grid 2: Blueprint for a new computing infrastructure I Foster, C Kesselman Elsevier	14096	2003
The Grid 2: Blueprint for a new computing infrastructure Elsevier	~~14089~~ *	2003
Grid: Blueprint for a New Computing Infrastructure (Elsevier Series in Grid Computing) C Kesselman, I Foster Morgan Kaufmann Publishers	~~13820~~ *	2004

Fig. 1. Example of scholar profile that includes title, affiliation, research interests, and publications.

Industry has developed interesting approaches e.g., LinkedIn [1] proposed a model for searching potential job candidates using deep representation learning based on talents search domain, skills entities, and talents feedback. HomeAdvisor [2] similarly helps users to find suitable handyman service personnel using their private database. Facebook [3] uniquely solved the cold-start issue by projecting users and events into the same latent space for matching heterogeneous information from different domains.

There have been limited works on identifying relevant scholars in academia using open datasets that include information such as scholar publications and funding records. Most existing scholar search and recommendation systems rely on querying and matching static information from online profiles, such as titles, affiliations, or declared research interests as shown in the Figure 1 example. Existing approaches typically use keyword matching [4] or NLP-based semantic analysis [5, 6] to assess a scholar's suitability for a given task.

However, such methods face inherent limitations. Research interest tags are often static and fail to reflect a scholar's evolving or interdisciplinary interests. Furthermore, these tags

Yuanxun Zhang is with Weee!, Fremont, CA 94536, USA.
E-mail: bigfishinriver@gmail.com
Xiyao Cheng is with University of Missouri-Columbia, Columbia, MO 65211, USA.
E-mail: xcheng@mail.missouri.edu
Roland Oruche is with University of Missouri-Columbia, Columbia, MO 65211, USA.
E-mail: rro2q2@mail.missouri.edu
Sai Swathi Sivarathri is with University of Missouri-Columbia, Columbia, MO 65211, USA.
E-mail: sivarathrisaiswathi@gmail.com
Prasad Calyam is with University of Missouri-Columbia, Columbia, MO 65211, USA.
E-mail: calyamp@missouri.edu

tend to be overly broad, lacking the granularity needed to match specific research tasks [7]. For instance, the scholar in Figure 1 might be matched only to general fields like "computer science," "distributed computing," or "data science." This overlooks the scholar's more specific focus on computational performance in distributed systems and cloud infrastructure. By exploring their publications, we also find contributions in areas like "big data" [8] and "machine learning," [9, 10] which are absent from their profile tags.

In our work, we propose a novel model viz., "ScholarFinder" to find suitable scholars using contextual (e.g., publications and funding record) information, who can successfully accomplish a set of given bold/multi-disciplinary research tasks across scientific domains. We present our ScholarFinder-VAE model in which we have applied the Variational Autoencoder (VAE) [11] for embedding scholar's knowledge based on their publication abstracts. The VAE model uses an encoder network to map the original dataset (publication abstracts) to the latent representation. In addition, it uses a decoder network to reconstruct the original dataset from the latent representation, and subsequently uses Stochastic Gradient Variational Bayes (SGVB) [12] to learn the model parameters and latent representations, which helps it to achieve better performance than state-of-the-art models. The advantage of VAE model is to understand the scholars' research distributions, which can help us to overcome the limitations in the length of scholars' descriptions. Using our model, each scholar can be mapped into a low dimensional latent space, and each dimension may represent the scholar's research areas or research interests. However, the embedding space generated by our VAE, as shown in Figure 2 using the t-SNE algorithm [13], does not clearly visualize clusters, making it difficult to differentiate scholars with different research interests. Ideally, if the embedding space were to form distinct clusters, with each cluster representing a research topic, it would significantly enhance downstream tasks such as clustering, visualization, prediction, and recommendation. Numerous studies [14] have shown that learning well-defined clusters can improve the performance of these downstream tasks by providing more structured and meaningful representations.

Consequently, taking inspiration from the Deep Embedding Cluster (DEC) model [14], we extend the ScholarFinder-VAE model by proposing the ScholarFinder-VDEC model that adds a clustering objective function to learn the features embedding and cluster assignment simultaneously. Unlike the DEC model that uses Autoencoder to learn the embeddings, we directly extend the VAE model by adding a clustering layer to learn the cluster assignment. We make such an addition due to the VAE yielding a better generalization performance over the Autoencoder. Further, we improve the generalization and clustering performance simultaneously by implementing the multi-tasks learning [15]. Consequently, our pre-trained knowledge embedding trained with our advanced VDEC model can be used for further downstream tasks. Our work can bridge the gap between generative modeling and clustering. Specifically, it shows how generative models can achieve clustering outcomes without significant alterations or dedicated clustering modules. Our dual functionality of generative modeling and clustering with improved performance sets our approach apart from classical generative or clustering methods.

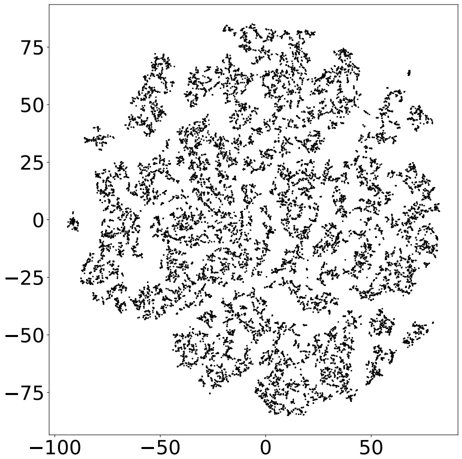


Fig. 2. Visualization of the latent space of scholars' knowledge embedding that is generated by the VAE model using the t-SNE algorithm.

Using a novel "follow-the-money" strategy, we apply our pre-trained knowledge embedding to a large collection of NSF (National Science Foundation) grant awards dataset [16] to predict whether a scholar is suitable for a particular grant or research task. This NSF grant award dataset is collected over the last twenty years and contains more than 20,000 award records (with project abstract and names), and 15,074 scholars who received grants. Using the above dataset in experimental studies, we evaluate our ScholarFinder-VDEC model with state-of-the-art models such as XGBoost, gradient boosting decision trees (GBDT), AdaBoost, Deep Neural Network (DNN), deep clustering models, such as DEC, VaDE, and recent large language models (LLMs), such as Bert4Rec [17] and OpenP5 [18] in terms of precision, recall, F1-score and accuracy metrics. Our clustering performance experiments aim to show how our embedding clusters similar scholars' interests and has generative characteristics to be able to sample new scholar recommendations from existing knowledge. As a case study, we also demonstrate how we can integrate our ScholarFinder as a pre-trained model in a knowledge graph to improve the downstream task performance. The contributions of our work can be summarized as follows:

- Enhanced the Deep Embedding Cluster (DEC) model by innovative incorporating Variational Autoencoder (VAE), achieving improved both representation learning and clustering performance.
- Boosted both generative and clustering performance by employing a novel multi-task learning to optimize generation and clustering simultaneously.
- Customized and adapted innovative negative sampling schemes to enhance downstream scholar task matching performance.
- To the best of our knowledge, first Designed and implemented the ScholarFinder system to identify and recommend suitable scholars for interdisciplinary research tasks.

The remainder of this paper is organized as follows: Section II discusses the related works. In Section III, we describe our methodology for the ScholarFinder model, in which we will describe our ScholarFinder-VAE and ScholarFinder-VDEC models. Section IV details the proposed ScholarFinder system architecture featuring a deep learning model. In Section V, we discuss our datasets used in experiments and ScholarFinder model evaluation experiment results. Section VI shows a case study to demonstrate about how ScholarFinder as a pre-trained model can be improved in terms of the downstream task performance. Section VII concludes the paper.

II. Related Work

Prior related works can be organized under three broad categories: (a) Recommendation system; (b) Deep representation learning; and (c) Embedding techniques.

Recommendation system. The recommendation system is considered to be one of the most successful approaches for personalized information filtering and searching schemes. Recommendation engines are an interesting alternative to search fields, as recommendation engines help users discover products or content that they may not come across otherwise. They are widely applied in E-commerce ecosystems, such as Amazon and eBay that recommend products to individual customers based on their interests or preferences. Traditional algorithms such as content-based filtering [19] and collaborative filtering [20] are commonly used recommendation algorithms and they have their limitations in accurately predicting user's ratings due to the sparsity of the datasets and cold-start issues.

Recently, advanced techniques have been proposed for solving the problem of cold start and sparsity and have become state-of-the-art approaches. For example, matrix factorization (MF) [21] decomposes rating matrix into the latent representation of users and items vectors, which allows making a rating prediction using the dot product of users and items vectors. Similarly, [22] used a probabilistic graphical model with latent users/items variables and observed ratings variables to perform matrix decomposition. Hence, latent representation learning became useful to understand the complex relationships in a high-dimensional dataset. [23] adapted a topic model to recommend collaborators. [24] proposed a probabilistic generative model to explore the expert behaviors and recommend experts in the collaborative networks by analyzing IBM ticket tracking logs. Similarly, Twitter's probabilistic model, *TWILITE* [25] is based on the Latent Dirichlet Allocation (LDA) [26] and recommends consumers' top-K tweets and users to read/follow, respectively. In addition, many recommendation systems currently use knowledge graphs to represent the relationships between graph nodes, along with deep representation learning to generate node embeddings [27, 28, 29].

Similar to prior works on recommender systems, our ScholarFinder uses embedding but in a contextual manner with unsupervised learning. In addition, our recommender system is the first to produce recommendations of suitable scholars based on their knowledge profiles in publications, as well as their publicly available funding records information.

Deep representation learning. Deep learning shows its best-in-class performance on problems that significantly outperform other solutions in multiple domains such as speech, language, vision. It involves training of a deep neural network using the back-propagation algorithm with a large amount of datasets. It can also effectively reduce the need for handcrafting feature engineering, which happens to be one of the most time-consuming parts in machine learning practice.

Representation learning is also an important area in deep learning that involves learning of the latent representation in high-dimensional data to extract useful features, such as basic components with image dataset, and semantic meaning with text dataset. Those features can be effectively used to perform classification, detection, as well as recommendation tasks. In earlier 1986, [30] presented a distributed representation learning of concepts with a simple neural network to learn family relationships. For a while, RBM and Autoencoder [31] were the popular deep generative models to model high-dimensional data for extracting features. Recently, VAE [12] and GAN [32] have shown impressive performance in generating data, which have made them become the most popular deep generative models. Autoencoder and VAE learn the latent representation (or embedding) through reconstruction of input data, and GAN learns the latent representation by playing games through generators and discriminators. In our work, we choose VAE to learn expertise embedding, because the documents can be treated as bag-of-words, which are not continuous. GAN is more suitable for use when given continuous datasets. In addition, VAE directly learns the distributions of the latent patterns that is more in line with our need to characterize scholars' distributions. Hence, in comparison with autoencoders and GAN, VAE can achieve better generalization performance by replacing latent variables with distributions, instead of discrete values. Recent works also try to improve autoencoder for a variety of downstream tasks by leveraging the methods of clustering [14] and graph neural networks [33, 34]. Among these methods, deep clustering methods [14, 35, 36] integrate feature learning and clustering, enhancing both processes. Unlike other methods with high computational and labeling costs, our approach scales linearly and supports large datasets and online scenarios. Consequently, the efficiency and accuracy of our approach are beneficial for effective data analysis and downstream tasks.

Recommendation systems also leverage deep learning techniques to achieve desired performance [37, 38]. The work in [39] proposed a restricted Boltzmann machine with one visible layer and one hidden layer to identify those users with similar interests. [40] combined convolutional neural network (CNN) and probabilistic matrix factorization (PMF) to capture contextual information of documents for improving performance. Similarly, the authors in [41] also fused matrix factorization (MF) and Multi-Layer Perceptron (MLP) to achieve better recommendation performance. Further, a deep matrix factorization (DMF) model was used in [42] to learn about both users and items in a common low dimensional space with non-linear projections. Further, deep representation learning is also applied in the context of knowledge graph based methods. The work in [43] proposes a novel representation learning for dynamic graphs based on the graph convolutional networks (GCNs), called DGCN, which achieves better performance in nodes clustering and link prediction. Similarly, the work in MI-KGNN [44] enhanced knowledge graph aware recommendations by proposing a multi-dimension interaction based attentional knowledge graph neural network.

In recent years, large language models (LLMs) have increasingly been explored for recommendation systems, leveraging their strong sequence modeling and generative capabilities. Traditional models like BERT4Rec [17] apply bidirectional Transformers (or Transformer Encoder) to capture user behavior sequences patterns and so as to predict future interactions. More recently, OpenP5 [18] provides an open platform for developing, training, and evaluating LLM-based generative recommenders, supporting both encoder-decoder (e.g., T5) and decoder-only (e.g., Llama-2) architectures. In our work, we use BERT4Rec and OpenP5 as baselines to evaluate the performance of our model on the ScholarFinder dataset.

Our ScholarFinder approach utilizes the VAE [12] deep generative model and the deep embedded clustering (DEC) [14] is utilized on top of VAE to capture scholars' knowledge embeddings through learning latent representation of given scholars' knowledge profile, i.e., the semantic meaning of data on the scholar's expertise. The DEC extension extracts latent representations from high dimensional scholar-related data in order to improve the downstream tasks (e.g., knowledge graph based recommendation systems) from our deep generative model.

Embedding techniques. Embedding is a popular technique that is widely used in dimension reduction [31] for finding good representations as well as visualizations [45]. Normally, we can use matrix factorization [21] and neural network based embedding [46] to learn embeddings. However, the former method performs a matrix decomposition that needs to be recomputed when a new user or item is added into the rating matrix. Neural network based embedding has become popular owing to its flexibility and incremental learning techniques.

Embedding is also useful in a recommendation system for solving the issue of data sparsity [47, 48]. Hence, various embedding methods are applied to a recommendation system: [49] proposed a skip-gram based model "prod2vec" to learn product embeddings for product-to-product predictions, and user embeddings for user-to-products predictions. [50] proposed a mixture embedding method for questions classification, which combines topic embeddings, word embeddings, and entity embeddings. [1] combined embedding and semantic representations for talent search at LinkedIn. [51] used a multi-modal embedding framework to provide more robust recommendations at Pinterest. [52] proposed a method that uses VAE to capture the latent representation of documents. [53] learned latent representation of scholars' knowledge in an unsupervised manner using the VAE model.

In addition, searches based on broad keywords can yield thousands of potential matches, making it difficult to identify the best fit. Scholars often work across multiple domains, and their research interests are not easily encapsulated by static tags. Thus, effective scholar discovery requires a dynamic approach that captures the nuanced and evolving nature of academic work [54]. A major challenge lies in developing an effective knowledge representation method to quantify scholars' expertise using contextual information, such as their publications, while covering a broad spectrum of research topics. Addressing this challenge is critical to overcoming the limitations of fixed research tags, which fail to capture the evolving and diverse nature of scholarly work [55]. Intuitively, a bag-of-words approach could represent publications as vectors. However, this method introduces significant issues such as sparsity and high dimensionality. Existing dimensionality reduction techniques like Principal Component Analysis (PCA) [56, 57] mitigate these challenges through linear transformations but are impacted with the computational overheads associated with processing high-dimensional matrices in large datasets. Other approaches, such as matrix factorization [21] or neural collaborative filtering [41] extract latent features through gradient descent to scale effectively to large datasets. Regardless, these methods focus solely on user/item indices and overlook contextual information. Consequently, whenever new data is added, re-computation or re-training of the model becomes necessary, and may impact practical application.

Inspired from word2vec [45], our ScholarFinder embeds scholars' knowledge in an unsupervised manner based on their knowledge profile. We assume that a scholar's expertise knowledge is similar to word embedding because of its corresponding semantic meaning of knowledge. Such a knowledge can be obtained via training on scholars' publications information. We evaluate this knowledge later along with scholars' publicly available funding information to recommend whether a scholar is suitable for particular research tasks or not. We also demonstrate how our model can be integrated into a knowledge graph for improving the performance for graph learning, and thereby improving the performance of downstream tasks such as prediction, clustering and visualization.

III. ScholarFinder Methodology

In this section, we detail our ScholarFinder methodology that can be organized under the following three aspects:

1) Learning good representation of scholars' knowledge using an embedding technique based on their publications. First, we will describe our ScholarFinder-VAE model that use variational autoencoder to learn latent knowledge representation; next, we will extend our ScholarFinder-VAE for learning the clustering features by demonstrating the ScholarFinder-VDEC model.
2) Demonstrating a novel negative sampling scheme for solving the issue of unbalanced labels in datasets.
3) Using our pre-trained knowledge embeddings to predict whether a scholar is suitable for a proposed task based on funding records in the NSF award dataset [16].

A. Knowledge Abstraction

The scholars' knowledge abstraction fully relies on scholars' publications. Scholars' publications are represented by a bag-of-words with a fixed size of the vocabulary C. These vocabulary are generated by using keywords from scientific topics taken from the ScienceDirect website [58]. In this way, we can visualize a scholar's expertise knowledge with a bag-of-words. As shown in Figure 3, we can easily recognize that this

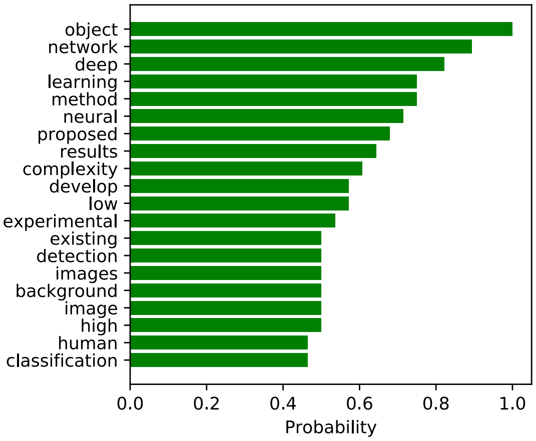


Fig. 3. Vector representation of bag-of-words based on a scholar's publications with top 20 frequent words shown above; word counts are normalized between 0 and 1.

particular scholar has expertise knowledge in deep learning from the most frequent words occurrence in his publications, which are normalized between 0 and 1.

Suppose we have M scholars, each scholar's publication in bag-of-words is denoted as x_i, which is a vector with the size of vocabulary C dimension. Then, all scholars' knowledge abstractions are represented by $\boldsymbol{X} \in \mathbb{R}^{M \times C}$. With scholars' knowledge abstraction, we will generate their embeddings.

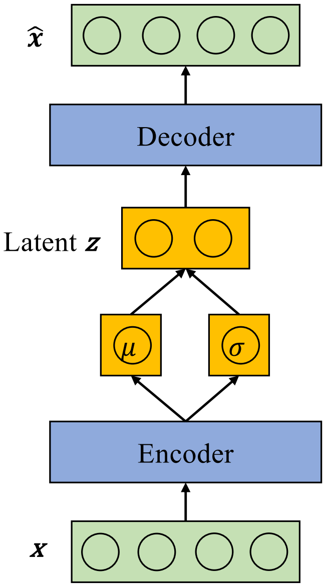


Fig. 4. ScholarFinder-VAE model network architecture based on variational autoencoder (VAE).

B. ScholarFinder-VAE: ScholarFinder using Variational Autoencoder

First, we describe the ScholarFinder-VAE model that uses variational autoencoder (VAE) to learn the latent representation for a scholar's knowledge profile based on his/her publications. Autoencoder [31] is typically a neural network that is trained by reconstructing input data through encoder/decoder networks in an unsupervised manner. In addition, a latent representation (coder) is also learned by reconstructing input data during the training process. The VAE model is similar to the Autoencoder, but instead of using discrete variables for latent representation in Autoencoder, VAE uses a distribution (such as, Gaussian distribution or other differentiable distribution) to present it. This in turn can help achieve better generalization performance. In order to infer the latent variables, VAE uses the Stochastic Gradient Variational Bayes (SGVB) with reparameterization trick [12].

In Figure 4, we present our VAE architecture for learning knowledge embedding. The goal of the VAE model is to learn knowledge (or latent) embedding $\mathbf{z}$ for each scholar based on the input of his/her publication x_i, and all scholars shared weights.

We define that the latent representation z_i is drawn from the Gaussian distribution,

$$z_i \sim \mathcal{N}\left(\mu_i, \sigma_i^2 \mathbf{I}\right) \tag{1}$$

With SGVB algorithm, the z_i can be approximated with,

$$z_i = \mu_i + \sigma_i \odot \epsilon \quad \text{and} \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}) \tag{2}$$

And z_i, μ_i, σ_i are hidden vectors with K dimension in VAE model shown in Figure 4 that are computed by a regular feed-forward neural network. The K is also a dimension of embedding, which needs to be defined. In our model, we use $K = 50$ for learning knowledge embedding. The *encoder network* and *decoder network* have two hidden layers, respectively. Each layer has 500 hundred neurons with ReLU [59] activation function. Thus, the l-th hidden layer h_l can be computed by,

$$h_l = \text{ReLu}(W_l^T h_{l-1} + b_l) \tag{3}$$

And, the μ, σ for i-th scholar can be calculated by,

$$\mu_i = W_\mu^T h_2 + b_\mu \tag{4}$$

$$\sigma_i = W_\sigma^T h_2 + b_\sigma \tag{5}$$

To learn weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ for obtaining knowledge embedding u_i, we need to solve the optimization problem by minimizing the reconstruction loss and KL-divergence loss. Hence, the VAE loss for i-th example is defined as,

$$\begin{aligned}\mathcal{L}(\boldsymbol{W}, x_i) \simeq \quad & - \quad \frac{1}{2}\sum_{k=1}^{K}\Big(1 + \log((\sigma_i^k)^2) - (\mu_i^k)^2 - (\sigma_i^k)^2\Big) \\ & - \quad \frac{1}{L}\log P(x_i \mid z_i, \boldsymbol{W}) \end{aligned} \tag{6}$$

This optimization problem can be solved using the stochastic gradient descent (SGD) or any of the other optimizers. In our work, we use the Adam optimizer [60].

After the training completion of our VAE model, we obtain knowledge embedding z_i for each scholar i using an encoder network. Then, we can use the scholars' pre-trained knowledge embedding to perform further proposed tasks.

C. ScholarFinder-VDEC: ScholarFinder using Variational Deep Embedded Clustering & Multi-Task Learning

As mentioned earlier in Section I, the drawback of the ScholarFinder-VAE model is the fact that the VAE model does not intend to separate the data in the latent space with different clusters based on the objective function in Equation (6), which may not achieve optimal performance in the downstream tasks. Hence, we have proposed our ScholarFinder-VDEC by improving the previous ScholarFinder-VAE model in the following two ways: (i) We added a clustering layer for learning clustering in the embedding space; (ii) We applied a multi-task learning scheme to learn autoencoding and clustering simultaneously.

1) Variational Deep Embedding Cluster Model for learning clustering in the embedding space: Inspired by the Deep Embedding Cluster (DEC) model [14], we added a clustering layer for learning clustering representation in embedding space, where we apply the method described in the DEC model. Different with DEC, we still use the variational autoencoder to learn the feature representation instead of using autoencoder, which can achieve better performance as per our earlier evaluation efforts [53]. There are three steps that are involved in learning the clustering features.

First, we initialize K cluster centroids, denoted as $\{\lambda_j\}_{j=1}^{K}$, in the embedding space z_i using the K-means algorithm. Each centroid λ_j is a vector with the same dimension as z_i. To obtain the initial values for λ_j, we train a variational autoencoder model for a few epochs, passing the data through the model to obtain embedded data points z_i. Finally, we apply the K-means clustering algorithm in the embedding space Z to determine the K initial centroids $\{\lambda_j\}_{j=1}^{K}$.

Second, we compute the soft assignment between embedded points z_i and λ_j with the t-distribution kernel function, which is defined as,

$$q_{ij} = \frac{\left(1 + \|z_i - \lambda_j\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}}{\sum_{j'} \left(1 + \|z_i - \lambda_{j'}\|^2 / \alpha\right)^{-\frac{\alpha+1}{2}}} \quad (7)$$

where the z_i is generated by the encoder network of VAE model, the α is the degrees of freedom of the t-distribution, which we set as $\alpha = 1$ in our experiments, and the q_{ij} is defined as the probability of assigning data point i to cluster j.

Third, we update the soft assignments by using an auxiliary target distribution. A common approach to measure similarity between the distributions P and Q is to use Kullback Leibler (KL) divergence [13]. More specifically, the model is trained by optimizing the difference between soft assignments q_{ij} and auxiliary target distribution p_{ij} using the KL divergence loss, which is defined as -

$$D_{KL}(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (8)$$

The choice of auxiliary target distribution p_{ij} is suggested by Xie *et al.*[14] as follows -

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}} \quad (9)$$

where $f_j = \sum_i q_{ij}$ are frequencies of soft assignments. This approach can be explained as a self-training strategy to learn the clustering assignment with a high confidence score.

2) Multi-Task learning: Although Xie *et al.* [14] mentioned that the DEC model simultaneously learns feature representations and cluster assignments, they use two individual phases to learn them separately. This approach may not find the optimal solution because it separately optimizes the two objective functions. However, many works [61, 62] investigate multi-tasking learning to optimize multiply objective functions simultaneously. Radford *et al.* [63] also found that jointly training with multiple objective functions help to improve the model generalization in their GPT models.

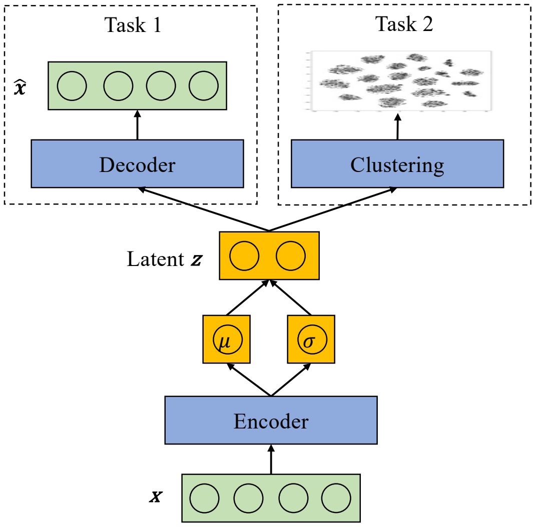


Fig. 5. ScholarFinder-VDEC model network architecture based on deep embedded clustering and multi-task learning.

As shown in Figure 5, we use multi-task learning scheme to learning feature representation (*Task 1*) and feature clustering (*Task 2*) simultaneously. The *Task 1* learns the latent representation $\mathbf{z}$ described in Section III-B; and the *Task 2* learns clustering feature mentioned in Section III-C1.

During the training of our VDEC model, we jointly optimize total loss $\mathbf{L}$ by combining the VAE loss L_1 defined in Equation (6) and KL loss L_2 defined in Equation (8). Specifically, we jointly optimize the following objective (with weight λ),

$$\mathbf{L} = L_1 + \lambda * L_2 \quad (10)$$

and the λ is hyperparameter that can be tuned based on the given objectives and dataset. In our experiment, we initially set a small λ during the early epochs to allow the model to focus on learning better feature representations with the VAE. As training progresses, we gradually increase λ to shift the model's focus towards learning the clustering features. The hyperparameter can be adjusted based on any given datasets and analysis objectives.

D. Negative Sampling

The negative sample issue is common in the most machine learning tasks. This is because it is easier to obtain a positive sample than a negative sample in the real world. For example, in job matching tasks, we can easily get datasets showing a person who got a job offer, but it is hard to obtain datasets indicating persons who rejected offers or failed in the interviews. We faced a similar missing negative samples issue in our dataset that only has positive samples. More specifically, in the NSF grant award datasets, we only have records that show that the scholars who got grants from NSF, but there are no records showing the scholars who failed to apply for a certain NSF grant.

Based on the above discussion, we need to generate negative samples for each scholar to train our downstream task model that predicts a score for a scholar given a task. Common methods such as Word2Vec [45] use random sampling to select a subset of dataset samples as negative samples. This approach works well when the dataset is large and spans a high-dimensional space, which naturally reduces correlations among samples. For example, in the case of Word2Vec, random sampling is likely to select a sample with a completely different semantic meaning from the target data. However, in our case, the manifold dimensions are low because we deal with a limited set of research fields that exhibit high correlations. For example, “computer systems” is closely related to “computer networking” and “cloud computing”, while “bioinformatics” has strong overlaps with “data mining” and “machine learning”.

The goal of our method is to sample adequate negative samples for each scholar. In the random sampling scheme, for each scholar, we randomly sample the same amount of negative samples with positive samples from whole datasets for indicating those NSF grants that are not suitable for that scholar. Given that the data correlation is high, this approach results in cases with a higher chance to sample an NSF grant that is similar or has some connection with a positive sample. This in turn makes the model training less effective for accurate predictions of scholar recommendations.

To address this issue, we introduce a novel negative sampling scheme that leverages our pre-trained scholar embedding technique. For each positive sample x_i, we compute its embedding vector z_i. We then randomly select 100 negative samples from the dataset and use the VAE model encoder to obtain their latent representations z_j. The Euclidean distance between z_i and each z_j is computed, and the distances are sorted in descending order. The final negative sample is selected from the top K samples with the largest distances. To determine the optimal number of negative samples, we conduct a negative sampling ratio analysis in Section V-C. Our experiments indicate that 20 negative samples are optimal for our datasets and model. This result also aligns with the recommendation from Word2Vec [45], which suggests 5–20 negative samples are sufficient for small datasets. Our negative sampling approach aims to maximize the difference between negative and positive samples by incorporating randomization, helping to avoid suboptimal solutions. This method resembles the negative sampling technique in Word2Vec, which has been proven to be effective [45].

This is allowed, because we pre-train the embeddings (knowledge embedding and tasks embeddings) separately in an unsupervised learning manner, and perform subsequent prediction tasks using the pre-trained embeddings. This approach cannot be applied to those methods when the tasks and knowledge embeddings are trained jointly (e.g., NCF [41], DSSM [64]). This is because the knowledge and task embeddings are in the same space. Task embeddings will have similar information in comparison with the knowledge embeddings.

E. Prediction with Pre-trained Knowledge Embedding

As we discussed in Section III-B, we use the ScholarFinder model (i.e., ScholoarFinder-VAE, ScholarFinder-VDEC) to get scholars’ embeddings and proposed tasks embedding separately. In this section, we will discuss how we can build the prediction downstream task using the pre-trained embeddings. For any scholar i and proposed task j, our goal is to predict whether the scholar i is suitable for the proposed task j based on their embeddings u_i, v_j. Basically, we pre-train the embedding models using ScholarFinder models with contextual information (i.e., publications and funding information). We separately trained the ScholarFinder models for generating scholars’ embedding and tasks’ embedding.

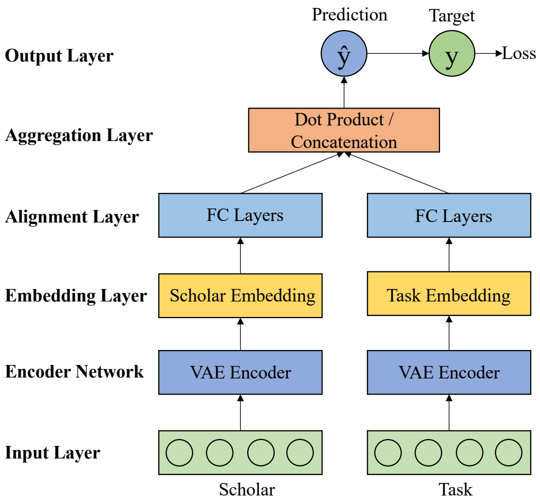


Fig. 6. Illustration showing the use of pre-trained knowledge embedding for proposed tasks prediction with two types aggregation options “Concatenation” or “Dot product”.

After obtaining our scholar embeddings u_i and task embedding v_j for a particular scholar or task using its own ScholarFinder encoder, we need to predict whether the scholar is suitable for a given research task or not. Intuitively, this problem looks easy to solve, which simply requires computing the dot product of these two vectors u_i, v_j, because we use the same embedding dimension K. This method can be used when they are trained with same VAE encoder. In our case, however, the two pre-trained embeddings u_i and v_j are learned separately with different ScholarFinder VAE encoder models, which are not in the same embedding space. Hence, the dot product computation will not work in this scenario. The reason why we separately trained the embedding model is that we want to uniquely train a model to generate scholar embedding, which can be applied to any of the types of downstream tasks.

To solve this problem, we firstly have the “Alignment Layer” on the top of the “Embedding Layer” as shown in Figure 6. The purpose of the “Alignment Layer” in our model is to align the two embedding spaces u_i and v_j before performing concatenation or a dot product on them. The “Alignment Layer” is constructed by two or three fully connected hidden layers with the ReLU activation function and 10% dropout [65]. If a scholar is suitable for a particular task, the weights in “Alignment Layer” will be learnt to align them into the same direction.

After “Alignment Layer”, we use an “Aggregation layer” to combine the two separate embeddings before output layer. We test two types of aggregation operations: “Concatenation” and “Dot Product” as shown in Figure 6. “Concatenation” is defined as an operation to concatenate two vectors $\mathbf{p}$ and $\mathbf{q}$ in the form of $[p_1, ..., p_n, q_1, ..., q_n]$; “Dot Product” is defined as inner dot product of two vectors $\mathbf{p}$ and $\mathbf{q}$ with $\mathbf{p} \cdot \mathbf{q}$.

Finally, the “Output Layer” model in Figure 6 will be connected to “Aggregation Layer” with a fully connected layer using *sigmoid* activation function for the output values between 0 to 1. To learn the model parameters, we need to minimize cross-entropy loss between prediction value $\hat{y}_{ij}$ and target value y_{ij} for all pairs of scholar i and task j,

$$\mathcal{L} = \sum_{i=1}^{M} \sum_{j=1}^{N} \Big[- y_{ij} \log \hat{y}_{ij} - (1 - y_{ij}) \log(1 - \hat{y}_{ij}) \Big] \tag{11}$$

In summary, our ScholarFinder model involves two stages: pre-train stage and prediction stage. In the pre-train stage, we need to pre-train scholars embeddings U and tasks embeddings V separately. Whereas, in the prediction stage, we check whether a particular scholar is suitable for a proposed research task given a particular scholar x_i, and a proposed task abstract y_j. The overall ScholarFinder model procedure above is summarized in Algorithm 1.

Algorithm 1 ScholarFinder model performs prediction with our pre-trained embeddings

```
(i). Pre-train Stage:
    (a) Given input scholars' publication in bag-of-words $X$, and input
        tasks' abstracts in bag-of-words $Y$
    (b) Pre-train the scholar embedding $U$ with a ScholarFinder-VDEC
        model;
    (c) Pre-train the task embedding $V$ with a ScholarFinder-VDEC
        model;
(ii). Prediction Stage:
    (a) Given a scholar input $x_i$ and a task input $y_i$;
    (b) Get its scholar embedding $u_i$ and task embedding $v_j$ with its
        VAE encoders
    (c) Use our prediction model (see Figure 6) to predict a score,
        which indicates whether a scholar is suitable for a given
        research task or not
```

IV. System Architecture

In this section, we present our proposed system architecture for the ScholarFinder model. As shown in Figure 7, there are five major components in our ScholarFinder system architecture: (i) Web Crawler; (ii) Data Pre-processing; (iii) Deep Learning model; and (iv) Visualization.

A. Web Crawler

In our system, the web crawler is used to extract publication abstracts from Google Scholar, which has the following steps shown in Figure 7: a) get all scholars names from NSF award dataset (Details of NSF award dataset will be discussed in Section V-A1); b) for each scholar, obtain his/her publications from Google Scholar using Google Scholar APIs [66]; c) for each publication, the web crawler will extract its abstract using Google Scholar APIs; d) save abstracts into a local storage system for later pre-processing.

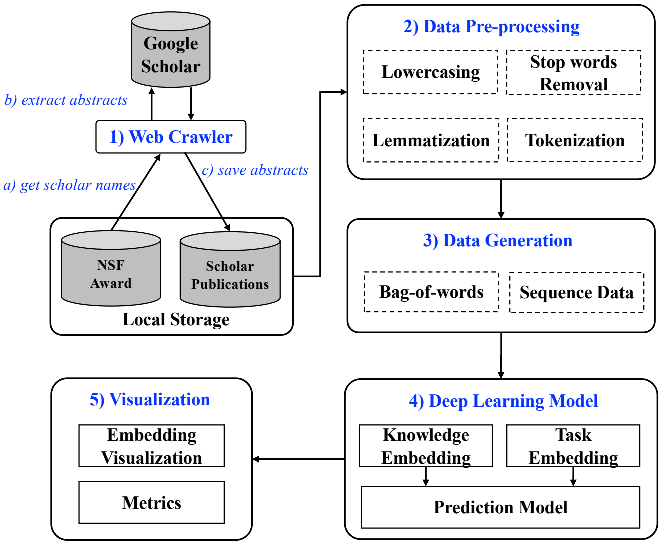


Fig. 7. System architecture of our ScholarFinder model.

B. Data Pre-processing

The goal of pre-processing is to generate bag-of-words for each abstract we collect. The basic steps for data pre-processing are: lowercasing the texts; removing the stop words that are not meaningful words, and significantly frequent (e.g., “the”, “a”) in texts; lemmatization to reduce or group different inflected forms of a word to a common base form; tokenization to tokenize words from; and bag-of-words to generate words counts for each abstract.

C. Data Generation

In the data generation stage, we use the text dataset to generate different data structures (bag-of-words or Sequence) based on the different scenarios. For example, we can use bag-of-words to understand basic topics of text, or use text sequence datasets for dialog design for questioning and answering.

D. Deep Learning Model

In our system, the deep learning model used is the one discussed in Section III which comprises of a “Knowledge Embedding” model that is used to train each scholar’s knowledge embedding based on his/her publication abstracts as bag-of-words representations which are processed in the previous stage. After finishing the training procedure, we obtain pre-trained knowledge embedding for each scholar, which can be used for future prediction, clustering, and visualization. In addition, a "Prediction" model is used to leverage the pre-trained knowledge embedding to perform another deep learning model. In our case, we use pre-trained knowledge embedding to predict whether a scholar is suitable for a proposed set of research tasks or not. We also show a case study in Section VI to demonstrate how we can apply the ScholarFinder to a Graph Neural Network (GNN) for improving performance of downstream tasks.

E. Visualization

The last part comprising of visualization and evaluation are used to: (a) visualize scholars' embedding in a 2D space, (b) for monitoring the model training performance (such as training loss or autoencoder reconstruction errors), and (c) for monitoring the evaluation performance in terms of precision, recall, F1-score, and accuracy metrics. We will demonstrate more details about visualization and evaluation in Section V-D (for performance evaluation), and Section V-E (for embedding visualization), respectively.

V. Performance Evaluation

In this section, we evaluate the precision, recall, F1-scores and accuracy of our model with state-of-the-art XGBoost and Deep Neural Network (DNN) models. Following this, we demonstrate visualization of embeddings based on our pre-trained knowledge embeddings to show the performance of clustering and generalization.

A. Datasets

In our work, we used two categories of open and large scholar related datasets: *NSF Awards* dataset, and *Publications* dataset, which cover most of the scientific fields. The *Publications* dataset is used to obtain scholars' knowledge embedding, and from the *NSF Awards* dataset, we extract NSF award abstract and scholars' names who successfully received competitive funding for checking whether a scholar is suitable for a research task or not.

1) NSF Awards: The NSF award dataset [16] mainly consists of categorical data related to abstracts by various authors who received NSF grants during the past twenty years. The award summaries consist of funded researchers' project profiles with keywords of research areas, tools, data sets, and research collaborators. The attributes of the dataset include abstract, title, award id, author details such as name, role as (PI) Principle investigator or Co-PI, institution information such as name, address, department, etc. We have collected 20,000 abstracts from the NSF award dataset that contains information that is relevant to 15,074 scholars who successfully obtained competitive funding from NSF grants over the past twenty years. Those scholars who received awards are considered as positive labels. Given the lack of negative samples in our dataset, we apply our negative sampling scheme discussed in Section III-D to generate the same amount of negative samples for training and testing activities. In order to compare performance of our negative sampling scheme, we consider a random negative sampling method as the competing solution.

2) Publications: As described in Section IV-A, the web crawler will extract all scholars' publications abstracts from all open publication archives and save them in our local storage system. Then, the collected abstracts are pre-processed by lowercasing, removing "stop words", tokenization discussed in Section IV-B Each abstract is represented as a "bag-of-words" in our model. Additionally for each author, we will use at most 10 recent papers abstracts in our model.

B. Experimental Setup

1) Metrics: We use *Precision, Recall, F1-Score, Accuracy* for our evaluation metrics:

- *Precision* is a metric to measure the ratio of correctly predicted positive labels (TP) to the total predicted positive labels (TP+FP).
$$Precision = \frac{TP}{TP + FP}$$
- *Recall* is the ratio of correctly predicted positive labels (TP) to the all labels (TP + FN) in actual class.
$$Recall = \frac{TP}{TP + FN}$$
- *F1 score* is the Harmonic mean of *Precision* and *Recall*, which consider the impact of both false positives (FP) and false negatives (FN). *F1 score* is usually more useful in the unbalanced dataset.
$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$
- *Accuracy* is metric to measure overall accuracy for both positive and negative labels
$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

2) Baselines: We consider state-of-the-art classical method sets including deep learning models and traditional machine learning models to evaluate our proposed model. Among the models we considered, DNN, XGBoost, GBDT, and AdaBoost are trained using bag-of-words directly and a random negative sampling scheme. In addition, we consider the DEC and VaDE state-of-the-art models that are trained similarly as our ScholarFinder using pre-trained embedding techniques and our novel negative sampling scheme.

- *DNN*: A deep neural network with 3 hidden layers is used in our evaluation experiments. Each layer uses ReLU activation function, and output layer uses sigmoid activation function with cross-entropy loss.
- *XGBoost* [67]: XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. XGBoost is based on Gradient Boosting algorithm and provides a parallel tree boosting (also known as GBDT, GBM) that solves many data science problems in a fast and accurate way.
- *GBDT*: Gradient boosting decision trees (GBDT) [68] is a machine learning technique that optimizes loss functions

to produce a prediction model in the form of decision trees. GBDT produces competitive and robust procedures for both regression and classification tasks.

- *AdaBoost*: Adaptive Boosting (AdaBoost) [69] boosts the performance of decision trees in classification programs by attempting to combine multiple weaker classifiers into one strong classifier. The weak learners that are refined based of their instances of misclassification of previous classifiers allow the algorithm to be adaptive.
- *DEC*: The Deep Embedded Clustering (DEC) model [14] learns clustering features for the embedding space through training with an autoencoder. As a pre-trained embedding model, we have applied our negative sampling method to the DEC model.
- *VaDE*: Variational Deep Embedding (VaDE) [70] employs an unsupervised approach for learning the latent space within the framework of a Variational Auto-Encoder (VAE). In contrast to DEC and our model, which incorporate clustering within the latent space, the VaDE model relies purely on the variational autoencoder algorithm.
- *BERT4Rec*: BERT4Rec (Bidirectional Encoder Representations from Transformers for Sequential Recommendation) [17] is a model that treats each user's shopping behavior as a sequence to predict the next possible items they may purchase. In our work, we adopt BERT4Rec to train a model using our ScholarFinder dataset.
- *OpenP5*: OpenP5 is an open-source platform designed as a resource to facilitate the development, training, and evaluation of LLM-based generative recommender systems [18]. The platform is implemented using encoder-decoder LLMs (e.g., T5) and decoder-only LLMs (e.g., Llama-2). Here we choose encoder-decoder LLMs as our baseline model, and apply our ScholarFinder dataset to show this model's performance.

3) ScholarFinder: As discussed in Section III, our ScholarFinder features two models (i.e., ScholarFinder-VAE and SchoalrFinder-VDEC) to learn the scholars' knowledge embedding. We have also proposed two types of aggregation layers ("Concatenation" and "Dot Product") for demonstrating the use of our pre-trained model to perform predictions. In the evaluation experiments, we evaluate both variants of the proposed ScholarFinder model, namely:

- *ScholarFinder-VAE (Concatenate)*: Use ScholarFinder-VAE for pre-training knowledge embedding and use "Concatenation" for prediction
- *ScholarFinder-VAE (Dot)*: Use ScholarFinder-VAE for pre-training knowledge embedding and use "Dot Product" for prediction
- *ScholarFinder-VDEC (Concatenate)*: Use ScholarFinder-VDEC for pre-training knowledge embedding and use "Concatenation" for prediction
- *ScholarFinder-VDEC (Dot)*: Use ScholarFinder-VDEC for pre-training knowledge embedding and use "Dot Product" for prediction

C. The Number of Negative Sample Selection Analysis

In Table II, we analyze the impact of varying the number of negative samples on our model's performance. Following the recommendations from the Word2Vec [45], which typically suggests using 3 to 20 negative samples, we adjust the number of negative samples to evaluate the model's performance. We define the number of negative samples as the ratio of positive to negative samples; for instance, a ratio of 5 means that for each positive example, 5 negative samples are sampled. The results in Table II show that our model's performance improves as the number of negative samples increases, achieving optimal performance when the number of negative samples reaches 20. Based on this result, we then use 20 negative samples for training and evaluation of our model.

D. Prediction Performance Evaluation Results

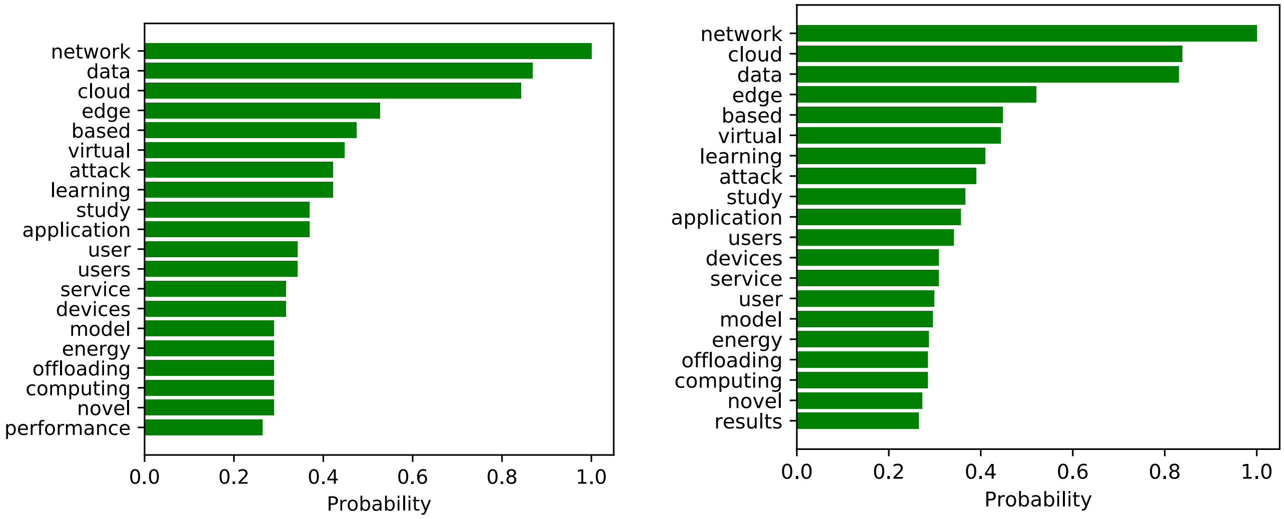
As shown in Table I, all our ScholarFinder models show better performance in comparison to the baseline models in terms of Precision, Recall, and F1-Score. Our ScholarFinder models achieve significantly better performance than other baseline models and improve the prediction performance by around 18% when considering all of the metrics. Particularly, the DEC and VaDE models are trained using the similar pre-trained techniques and novel negative sampling scheme, which proves that our ScholarFinder model has better generalization performance than the DEC and VaDE models. More specifically, VaDE showed the worst performance compared to other models. In our experiments, we observed that the training loss converged quickly after a few epochs, and there was no further improvement in performance. We also adjusted various hyperparameters, however these efforts did not yield any further improvement. We conclude that the original VaDE model may need deeper optimization for our dataset.

In addition, we compare our model with LLM-based recommendation models, including BERT4Rec, a sequential recommendation model using bidirectional Transformers, and OpenP5, a generative recommender system built on encoder-decoder LLMs. Our model outperforms all baselines on our scholar dataset, as it incorporates more scholar-specific information, such as publication records, enhancing its effectiveness in this domain. This finding demonstrates that our model surpasses other state-of-the-art models in the academic social network recommendation domain. Compared to these baseline models, our ScholarFinder-VDEC model achieves slightly better performance than our previous ScholarFinder-VAE model, with around a 3% improvement. Moreover, ScholarFinder-VDEC proves particularly useful for supporting downstream clustering and classification tasks, without compromising overall model performance. We also evaluated the clustering performance to prove the advantage of using the ScholarFinder-VDEC. In addition, for the prediction model, the "concatenation model" achieves slightly better performance than the "dot product model". Hence, among the ScholarFinder model variants, the ScholarFinder-VDEC (concatenation) layer clearly achieves the best performance.

Our ScholarFinder model is a generative model designed for high-quality data generation through learning of the rep-

TABLE I
EVALUATION RESULTS OF OUR PROPOSED SCHOLARFINDER MODELS COMPARISON WITH STATE-OF-THE-ART MODELS IN TERMS OF PRECISION, RECALL, F1 SCORE AND ACCURACY METRICS.

Model	Accuracy	Precision	Recall	F1-score
AdaBoost	0.51	0.51	0.51	0.51
GBDT	0.59	0.59	0.59	0.59
XGBoost	0.59	0.59	0.59	0.59
DNN	0.73	0.73	0.73	0.72
Bert4Rec	0.81	0.86	0.81	0.80
OpenP5	0.67	0.67	0.67	0.67
VaDE (Concatenate)	0.50	0.53	0.50	0.33
VaDE (Dot)	0.50	0.25	0.50	0.33
DEC (Concatenate)	0.76	0.77	0.76	0.76
DEC (Dot)	0.76	0.77	0.76	0.76
ScholarFinder-VAE (Concatenate)	0.96	0.96	0.96	0.96
ScholarFinder-VAE (Dot)	0.94	0.94	0.94	0.94
ScholarFinder-VDEC (Concatenate)	**0.99**	**0.99**	**0.99**	**0.99**
ScholarFinder-VDEC (Dot)	0.96	0.96	0.96	0.96



(a) Input of a normalized bag-of-words representation of a scholar's publications with top 20 frequent words shown above.

(b) Reconstruction of a normalized bag-of-words representation of the scholar's publications.

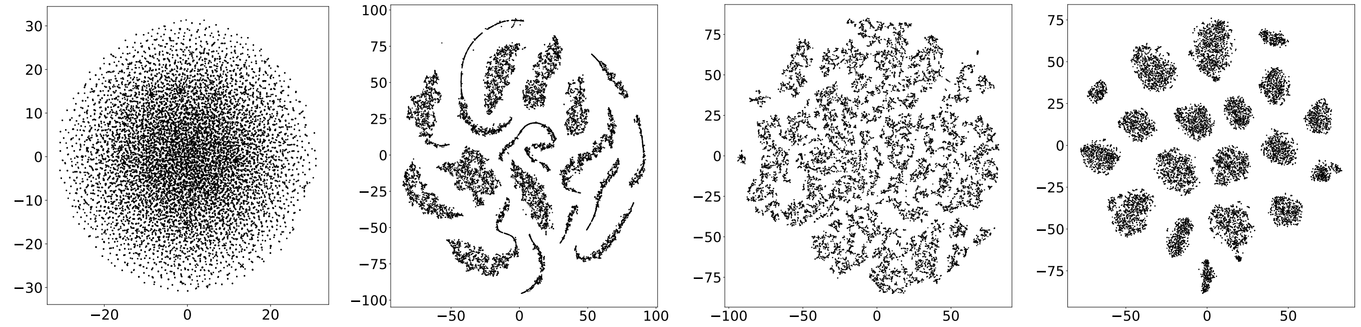
Fig. 8. Reconstruction performance results of knowledge embedding using ScholarFinder-VDEC model.

TABLE II
IMPACT OF THE NUMBER OF NEGATIVE SAMPLES ON THE PERFORMANCE OF SCHOLARFINDER MODELS IN TERMS OF ACCURACY SCORE.

Negative Sample Number	**1**	**5**	**10**	**20**
Accuracy	0.91	0.97	0.98	0.99

resentation of latent space, which can directly enhance the performance of downstream tasks. Additionally, ScholarFinder prioritizes the generation of well-structured clusters. As noted earlier, clustering improves downstream tasks by organizing data into coherent groups, reducing noise, and revealing hidden patterns. These well-defined clusters make the data more interpretable and meaningful, enhancing the quality and effectiveness of subsequent analyses. To evaluate clustering performance quantitatively, we employ two unsupervised metrics: (i) the *Calinski-Harabaz Index* (CHI) [71] and (ii) the *Silhouette Coefficient* [72]. These metrics are vital for assessing how effectively the clustering algorithm groups similar data points and separates dissimilar ones.

The Calinski-Harabaz Index calculates the ratio of the dispersion between clusters to the dispersion within clusters, with dispersion defined as the sum of squared distances. A higher CHI score reflects clusters that are compact and well-separated, which ensures that the resulting groups are distinct and easy to interpret, enhancing data clarity for downstream analysis. The Silhouette Coefficient further measures clustering quality by considering both intra-cluster distance and the nearest inter-cluster distance, generating a score from -1 to 1. A positive score near 1 indicates well-separated clusters, promoting clearer differentiation of data points, while negative



(a) VaDE (b) DEC (c) ScholarFinder-VAE (d) ScholarFinfder-VDEC

Fig. 9. Visual comparison of the latent space of scholars' knowledge embedding as generated by different models using the t-SNE algorithm.

TABLE III
CLUSTERING PERFORMANCE EVALUATION USING CALINSKI-HARABAZ INDEX (CHI) AND SILHOUETTE COEFFICIENT METRICS FOR VARYING N, WHICH DENOTES THE NUMBER OF CLUSTERS

N	ScholarFinder-VAE		DEC		ScholarFinder-VDEC		VaDE	
	CHI	**Silhouette**	**CHI**	**Silhouette**	**CHI**	**Silhouette**	**CHI**	**Silhouette**
5	13014.2	0.35	17201.63	0.44	**30297.54**	**0.52**	10020.20	0.31
10	14432.81	0.38	18592.40	0.48	**33940.47**	**0.55**	11691.97	0.33
15	14637.80	0.37	21443.83	0.49	**36266.32**	**0.56**	11851.55	0.32
20	15157.36	0.38	17714.23	0.45	**32867.33**	**0.50**	11804.71	0.32
25	15774.73	0.39	16988.65	0.44	**34347.43**	**0.54**	11639.60	0.32
30	15898.09	0.38	16857.02	0.43	**29656.65**	**0.48**	11133.99	0.32

scores highlight poor clustering performance, which can hinder subsequent tasks by introducing ambiguity and errors.

We compare the performance of Calinski-Harabaz Index (CHI) and Silhouette Coefficient metrics with different number of clusters N in order to evaluate of the effectiveness of our model. Table III shows the results; note that in case of both the metrics, a higher score indicates better performance and each score is calculated by averaging the scores from 5 tests. Given that the Scholar-VAE, DEC, VaDE and ScholarFinder-VDEC models have randomization process in the initialization, each case is run 5 times to get the average values. We can observe that our ScholarFinder-VDEC model has much better clustering performance than other models in the various cluster number scenarios. This observation signifies that we achieve better data grouping and clearer separation of clusters or distinct and meaningful clusters, enhancing the quality and interpretability of subsequent analyses.

E. Embedding Performance Evaluation

One goal in our ScholarFinder model is to better learn the clustering features that are helpful for necessary classification and clustering visualization tasks. In this section, we evaluate visualization performance in the embedding space among the DEC, and ScholarFinder models. Among these models, we use the same hyperparameters, such as the batch size, the learning rate, and the number of epoch to train the embedding. We use the same number of clusters (K=20) for DEC and ScholarFinder-VDEC. As shown in Figure 9, our previous ScholarFinder-VAE and VaDE models cannot learn the clustering features on our NSF dataset. DEC can learn the good clustering features as shown in the evaluation results with 20 clusters. However, our ScholarFinder-VDEC model achieves a better clustering performance in comparison with the DEC model. The reason being, the gaps among clusters are clearer and the shapes of clusters look more regular than the DEC model, which also validated the scores mentioned in Table III.

In addition, we can also evaluate generalization performance of the embedding space by visualizations of the reconstruction process. As shown in Figure 8, we can see that our ScholarFinder-VDEC can basically capture the scholar's research areas (such as cloud computing, networking) through the reconstruction results.

We also explored another interesting phenomenon in our experiments that involves a generative nature to query scholars based on their particular expertise knowledge or cross-domain knowledge. For example, let us assume we want to find a scholar who can connect the areas between "Networking and Cloud" and "Machine Learning and Data Mining" domains. Then, we can just find a known scholar in "Networking and Cloud" domain and a known scholar in "Machine Learning and Data Mining" domain. Then, using the ScholarFinder-VDEC encoder network, we can get the embeddings for these two scholars u_i and u_j respectively, and then compute the middle point $u_k = (u_i + u_j)/2$ in the embedding space. Finally, we just need to search the nearest neighbor points in the embedding space to look up the scholars we are querying.

As shown in Figure 10, we can visualize the scholar's expertise knowledge by using the ScholarFinder-VDEC decoder network. We found that the scholar in this case has expertise

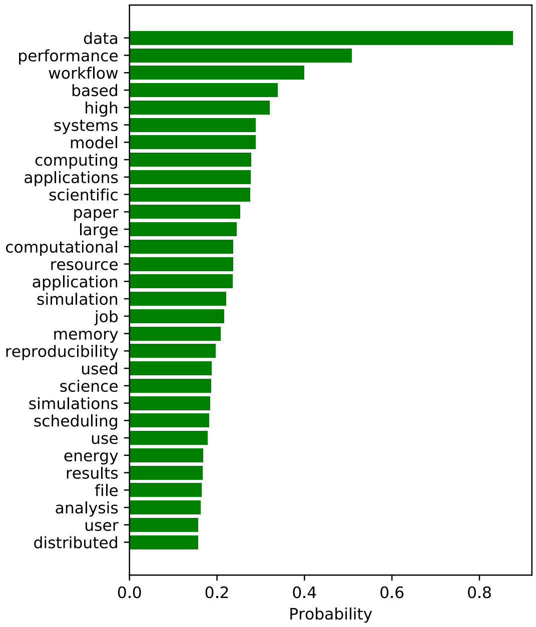


Fig. 10. Sample a new scholar obtained by fusing a scholar from networking and cloud domains, and a scholar from machine learning and data mining domains.

knowledge in the area of "data computation", "workflow performance" or "distributed system", which are really synergistic and bridge areas to connect between "Networking and Cloud" and "Machine Learning and Data Mining" domains. We can thus validate that our ScholarFinder has generative characteristics given the fact many emerging collaborations among these two domains are increasingly focused on building efficient infrastructures for data computing, machine learning, and data analysis workflows.

VI. Knowledge Graph Integration Case Study

In this section, we will discuss how we can use our ScholarFinder model with pre-trained embedding to improve the performance of a Graph Neural Network (GNN), such as GraphSAGE [73].

A. Building a Knowledge Graph using the NSF Dataset

Our NSF dataset includes entities such as scholars and awards information. Based on that information, we build a knowledge graph involving entities (such as scholars and awards) and relationships between entities. The relationship indicates how a specific scholar has obtained funding from NSF. This type of graph is called a Heterogeneous Knowledge Graph. As shown in Figure 11, in order to simply apply the GraphSAGE algorithm, we ignore the award entity and only keep the scholar entity. Following this, we build edges among scholars for those working on the same NSF grant awards. Finally, we can transform the Heterogeneous Knowledge Graph into a Homogeneous Knowledge Graph. We extract 1,174 scholars from our NSF award dataset to build our knowledge graph. For nodes' label, we use the awards' division name as the scholar's node labels (such as natural science, society science). Thereby, we can map each scholar into one class for classification tasks.

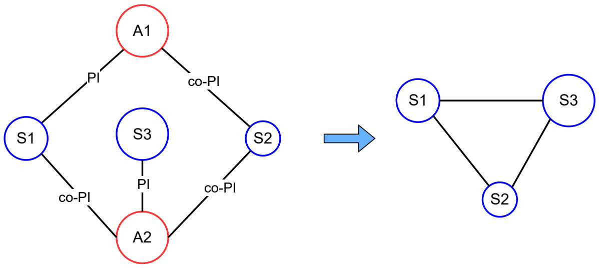


Fig. 11. An example of NSF grant Knowledge Graph, which can be transformed from Heterogeneous Knowledge Graph (left) to Homogeneous Knowledge Graph (right). The "S1, S2, S3" denotes scholars, and the "A1, A2" denotes NSF grant awards.

B. Learning with GraphSAGE

We have proved that our pre-trained ScholarFinder model can improve simple downstream tasks in the previous sections. Herein, we demonstrate how our model can be applied to the area of Graph Neural Networks (GNNs) to further improve their performance. In our case study, we consider a classical graph algorithm GraphSAGE [73], which is a representation learning technique suitable for dynamic graphs. It learns nodes embedding by aggregating information from a node's local neighborhood using a neural network.

In addition, GraphSAGE allows us to leverage nodes' features (e.g., text, image features) to efficiently generate node embeddings. This approach is more advanced than the previous classical model node2vec [45] that only uses index information. Hence, we initialize the nodes' features in GraphSAGE with our pre-trained scholar embeddings to evaluate whether our model can improve the graph learning performance or not.

C. Performance Evaluation

GraphSAGE can be trained in either a supervised or unsupervised manner. For the purposes of our case study, we trained a supervised GraphSAGE for node classification. We were able to map each scholar into one class based on the category of NSF award, as we mentioned earlier. To evaluate the performance of our pre-trained ScholarFinder model, we trained one GraphSAGE model using our pre-trained ScholarFinder-VEDC embedding, and another one without our pre-trained ScholarFinder embedding.

We use *precision*, *recall*, and *f1 score* as evaluation metrics to analyze our experiment results. As shown in Table IV, the GraphSAGE-S, which uses our pre-trained *ScholarFinder-VEDC* scholars' embeddings achieves much better performance than the original GraphSAGE in every metric.

VII. Conclusion And Future Work

In this paper, we proposed a novel ScholarFinder that features two models (i.e., ScholarFinder-VAE and ScholarFinder-VDEC) to find scholars who are suitable for specific research tasks that require expertise in multiple domains. Our ScholarFinder-VAE model uses Variational Autoencoder (VAE), to embed a scholar's expertise knowledge based on his/her publications, and each scholar is represented

TABLE IV
GRAPHSAGE MODEL PERFORMANCE EVALUATION USING PRECISION, RECALL AND F1 METRICS WITH DIFFERENT PRE-TRAINED EMBEDDING.

Models	Precision	Recall	F1
GraphSAGE	42.4	48.1	39.3
GraphSAGE-S	**55.6**	**52.9**	**60.7**

with latent knowledge representations. These knowledge representations can be used for matching similar interests for collaborations, or to evaluate whether an individual scholar is qualified to perform particular research tasks. We showed how our ScholarFinder-VDEC model extends the ScholarFinder-VAE model by adding a deep embedded clustering method to learn the clustering assignments. We have also applied the multi-tasks learning to our ScholarFinder-VDEC to improve its generalization and clustering performance. We have shown how our ScholarFinder methodology uses pre-trained knowledge embedding to build further downstream prediction and clustering tasks by proposing two different DNN models (i.e., Concatenation model and Dot Product model). Leveraging the pre-trained embedding scheme, we also proposed a novel negative sampling scheme that solves the issue of unbalanced labels in our scholars' funding record dataset.

In our model evaluation experiments, we compared the ScholarFinder model variants with state-of-the-art baseline models (i.e., XGBoost, GBDT, AdaBoost, DNN, GraphSAGE, DEC, VaDE) and recent LLM based models (i.e., Bert4Rec, OpenP5). Our evaluation results showed that our ScholarFinder models variants consistently can achieve at least 18% better performance in terms of precision, recall, F1-score and accuracy. In addition, our clustering visualization performance results showed that our embedding can cluster those scholars with similar research interests and has generative characteristics to be able to sample new scholar recommendations from existing knowledge. These experimental results also proved that our ScholarFinder with variational autoencoder (VAE) is good at capturing the latent representation of scholars' knowledge to improve the performance of downstream tasks; and the deep embedded clustering algorithm can achieve better clustering performance when it is trained with VAE model using multi-task learning techniques. Lastly, we demonstrated that our novel negative sampling scheme can significantly improve the performance when facing the issue of imbalanced datasets. In addition, we also use a case study to demonstrate how graph learning can leverage our ScholarFinder to further improve the model performance.

Possible future directions for this work include building visualization interfaces in our ScholarFinder to browse and drill-down knowledge patterns. In addition, one can investigate the efficacy of using knowledge graphs and GNNs on the NSF grant dataset that enables fusing of heterogeneous information (such as co-authorships, institutes) for identifying groups/teams of scholars that are suitable for collaboration for a given research task. Our ScholarFinder could also be integrated with a chatbot for recommending scholars to solve research tasks in emerging areas such as precision medicine, data-driven agriculture and autonomous materials design.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation grant numbers OAC-1730655 and OAC-2006816. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] R. Ramanath, H. Inan, G. Polatkan, B. Hu, Q. Guo, C. Ozcaglar, X. Wu, K. Kenthapadi, and S. C. Geyik, "Towards deep and representation learning for talent search at linkedin," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 2253–2261.

[2] HomeAdvisor, "Homeadvisor is a digital marketplace formerly known as servicemagic." 1998. [Online]. Available: https://www.homeadvisor.com/

[3] L. Tang and E. Y. Liu, "Joint user-entity representation learning for event recommendation in social network," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 271–280.

[4] J. Sun, J. Ma, X. Cheng, Z. Liu, and X. Cao, "Finding an expert: A model recommendation system," 2013.

[5] H. H. Lathabai, A. Nandy, and V. K. Singh, "Institutional collaboration recommendation: An expertise-based framework using nlp and network analysis," *Expert Systems with Applications*, vol. 209, p. 118317, 2022.

[6] M. I. Hossain, S. Kobourov, H. Purchase, and M. Surdeanu, "Rematch: Research expert matching system," in *2018 International Symposium on Big Data Visual and Immersive Analytics (BDVA)*. IEEE, 2018, pp. 1–10.

[7] M. G. Armentano, D. Godoy, M. Campo, and A. Amandi, "Nlp-based faceted search: Experience in the development of a science and technology search engine," *Expert systems with applications*, vol. 41, no. 6, pp. 2886–2896, 2014.

[8] I. Foster, R. Ghani, R. S. Jarmin, F. Kreuter, and J. Lane, *Big data and social science: A practical guide to methods and tools*. Chapman and Hall/CRC, 2016.

[9] S. Sahoo, T. Russo, J. Elliott, and I. Foster, "Machine learning algorithms for modeling groundwater level changes in agricultural regions of the us water resour res 53: 3878–3895," 2017.

[10] B. Blaiszik, L. Ward, M. Schwarting, J. Gaff, R. Chard, D. Pike, K. Chard, and I. Foster, "A data ecosystem to support machine learning in materials science," *MRS Communications*, vol. 9, no. 4, pp. 1125–1133, 2019.

[11] D. P. Kingma, M. Welling *et al.*, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.

[12] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[13] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[14] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, 2016, pp. 478–487.

[15] S. Ruder, "An overview of multi-task learning in deep neural networks," *arXiv preprint arXiv:1706.05098*, 2017.

[16] NSF, "National science foundation." 1950. [Online]. Available: https://nsf.gov/awardsearch/

[17] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 1441–1450.

[18] S. Xu, W. Hua, and Y. Zhang, "Openp5: An open-source platform for developing, training, and evaluating llm-based recommender systems," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024, pp. 386–394.

[19] B. Krulwich and C. Burkey, "Learning user information interests through extraction of semantically significant phrases," in *Proceedings of the AAAI spring symposium on machine learning in information access*, vol. 25, no. 27. Palo Alto, California, 1996, p. 110.

[20] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, ser. CSCW '94. New York, NY, USA: ACM, 1994, pp. 175–186. [Online]. Available: http://doi.acm.org/10.1145/192844.192905

[21] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[22] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Advances in neural information processing systems*, 2008, pp. 1257–1264.

[23] J. Tang, S. Wu, J. Sun, and H. Su, "Cross-domain collaboration recommendation," in *KDD'12*. New York, NY, USA: ACM, 2012, pp. 1285–1293. [Online]. Available: http://doi.acm.org/10.1145/2339530.2339730

[24] H. Sun, M. Srivatsa, S. Tan, Y. Li, L. M. Kaplan, S. Tao, and X. Yan, "Analyzing expert behaviors in collaborative networks," in *KDD'14*. New York, NY, USA: ACM, 2014, pp. 1486–1495. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623722

[25] Y. Kim and K. Shim, "Twilite: A recommendation system for twitter using a probabilistic model based on latent dirichlet allocation," *Information Systems*, vol. 42, pp. 59–77, 2014.

[26] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[27] M. Kherad and A. J. Bidgoly, "Recommendation system using a deep learning and graph analysis approach," *Computational Intelligence*, vol. 38, no. 5, pp. 1859–1883, 2022.

[28] L. Yu, L. Sun, B. Du, C. Liu, W. Lv, and H. Xiong, "Heterogeneous graph representation learning with relation awareness," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[29] C. Gao, X. Wang, X. He, and Y. Li, "Graph neural networks for recommender system," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 2022, pp. 1623–1625.

[30] H. Geoffrey H., "Learning distributed representations of concepts," in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, 1986.

[31] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[32] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[33] S. Xiao, S. Wang, and W. Guo, "Sgae: Stacked graph autoencoder for deep clustering," *IEEE Transactions on Big Data*, vol. 9, no. 1, pp. 254–266, 2022.

[34] K. Guo, Z. Chen, X. Lin, L. Wu, Z.-H. Zhan, Y. Chen, and W. Guo, "Community detection based on multiobjective particle swarm optimization and graph attention variational autoencoder," *IEEE Transactions on Big Data*, 2022.

[35] J. Cai, J. Fan, W. Guo, S. Wang, Y. Zhang, and Z. Zhang, "Efficient deep embedded subspace clustering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1–10.

[36] J. Xu, Y. Ren, H. Tang, X. Pu, X. Zhu, M. Zeng, and L. He, "Multi-vae: Learning disentangled view-common and view-peculiar visual representations for multi-view clustering," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9234–9243.

[37] A. Da'u, N. Salim, and R. Idris, "Multi-level attentive deep user-item representation learning for recommendation system," *Neurocomputing*, vol. 433, pp. 119–130, 2021.

[38] J. Ni, Z. Huang, J. Cheng, and S. Gao, "An effective recommendation model based on deep representation learning," *Information Sciences*, vol. 542, pp. 324–342, 2021.

[39] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 791–798.

[40] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-

aware recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 2016, pp. 233–240.

[41] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[42] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems." in *IJCAI*, vol. 17. Melbourne, Australia, 2017, pp. 3203–3209.

[43] C. Gao, J. Zhu, F. Zhang, Z. Wang, and X. Li, "A novel representation learning for dynamic graphs based on graph convolutional networks," *IEEE Transactions on Cybernetics*, 2022.

[44] Z. Wang, Z. Wang, X. Li, Z. Yu, B. Guo, L. Chen, and X. Zhou, "Exploring multi-dimension user-item interactions with attentional knowledge graph neural networks for recommendation," *IEEE Transactions on Big Data*, 2022.

[45] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[46] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[47] X. Hu, J. Xu, W. Wang, Z. Li, and A. Liu, "A graph embedding based model for fine-grained poi recommendation," *Neurocomputing*, vol. 428, pp. 376–384, 2021.

[48] S. Yan, H. Wang, Y. Li, Y. Zheng, and L. Han, "Attention-aware metapath-based network embedding for hin based recommendation," *Expert Systems with Applications*, vol. 174, p. 114601, 2021.

[49] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp, "E-commerce in your inbox: Product recommendations at scale," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1809–1818.

[50] D. Li, J. Zhang, and P. Li, "Representation learning for question classification via topic sparse autoencoder and entity embedding," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 126–133.

[51] A. Pal, C. Eksombatchai, Y. Zhou, B. Zhao, C. Rosenberg, and J. Leskovec, "Pinnersage: Multi-modal user embedding framework for recommendations at pinterest," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 2311–2320.

[52] Y. Miao, L. Yu, and P. Blunsom, "Neural variational inference for text processing," in *International conference on machine learning*, 2016, pp. 1727–1736.

[53] Y. Zhang, S. S. Sivarathri, and P. Calyam, "Scholarfinder: knowledge embedding based recommendations using a deep generative model," in *2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService)*. IEEE, 2020, pp. 88–95.

[54] X. Cheng, L. S. Edara, Y. Zhang, M. Kejriwal, and P. Calyam, "Influence role recognition and llm-based scholar recommendation in academic social networks," in *2024 IEEE 11th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2024, pp. 1–11.

[55] X. Cheng, Y. Zhang, H. Joshi, M. Kejriwal, and P. Calyam, "Knowledge graph-based embedding for connecting scholars in academic social networks," in *2023 IEEE 10th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2023, pp. 1–10.

[56] V. Gupta, S. Giesselbach, S. Rüping, and C. Bauckhage, "Improving word embeddings using kernel PCA," in *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, I. Augenstein, S. Gella, S. Ruder, K. Kann, B. Can, J. Welbl, A. Conneau, X. Ren, and M. Rei, Eds. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 200–208. [Online]. Available: https://aclanthology.org/W19-4323

[57] B. M. S. Hasan and A. M. Abdulazeez, "A review of principal component analysis algorithm for dimensionality reduction," *Journal of Soft Computing and Data Mining*, vol. 2, no. 1, pp. 20–30, 2021.

[58] ScienceDirect, "A large bibliographic database of scientific and medical publications of the british publisher elsevier." 1997. [Online]. Available: https://www.sciencedirect.com/topics/index

[59] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 315–323.

[60] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[61] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 1930–1939.

[62] X. Ma, L. Zhao, G. Huang, Z. Wang, Z. Hu, X. Zhu, and K. Gai, "Entire space multi-task model: An effective approach for estimating post-click conversion rate," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 1137–1140.

[63] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training," 2018.

[64] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 2333–2338.

[65] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of*

machine learning research, vol. 15, no. 1, pp. 1929–1958, 2014.

[66] S. A. Cholewiak, "Google scholar apis," 2014. [Online]. Available: https://pypi.org/project/scholarly/

[67] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM, 2016, pp. 785–794.

[68] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

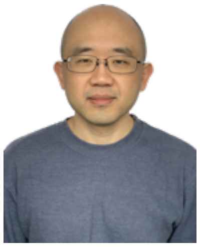
[69] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[70] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: an unsupervised and generative approach to clustering," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, ser. IJCAI'17. AAAI Press, 2017, p. 1965–1972.

[71] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[72] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[73] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.



Yuanxun Zhang received his BE degree from Southwest Jiaotong University, China, in 2006 and Ph.D degree in Computer Science from the University of Missouri-Columbia, MO, USA, and Computer Science at the University of Missouri-Columbia in 2021. He is interested in the theory and practice of understanding and modeling complex big data for making better decisions, for solving these problems involving information retrieval, recommendation system, and machine learning.

Xiyao Cheng received her MS degree in Computer Science from China Agriculture University. She is currently pursuing Ph.D. degree in the Department of Electrical Engineering and Computer Science at University of Missouri-Columbia. Her current research interests include knowledge graph based data modeling and artificial intelligence. She is a student member of IEEE.

Roland Oruche received his BS in Information Technology from the University of Missouri-Columbia. He is currently pursuing his Ph.D. in Computer Science at the University of Missouri-Columbia. His research interests include Machine Learning, Natural Language Processing, Human-Centered AI, and Dialog Systems.

Sai Swathi Sivarathri received B.Tech degree from JNTU Hyderabad, India, in 2014 and M.S degree in computer engineering from the University of Missouri-Columbia, MO, USA, in 2019. Her research interests include Cloud Computing, Machine Learning and Big Data Analytics.

Prasad Calyam received his MS and Ph.D. degrees from the Department of Electrical and Computer Engineering at The Ohio State University in 2002 and 2007, respectively. He is currently a Professor in the Department of Computer Science at University of Missouri-Columbia. Previously, he was a Research Director at the Ohio Supercomputer Center. His research interests include distributed and cloud computing, computer networking, and cyber security. He is a Senior Member of IEEE.