A Variational Neural Architecture for Skill-based Team Formation

RADIN HAMIDI RAD, Toronto Metropolitan University, Canada
HOSSEIN FANI, University of Windsor, Canada
EBRAHIM BAGHERI and MEHDI KARGAR, Toronto Metropolitan University, Canada
DIVESH SRIVASTAVA, AT&T Chief Data Office, USA
JAROSLAW SZLICHTA, York University, Canada

Team formation is concerned with the identification of a group of experts who have a high likelihood of effectively collaborating with each other to satisfy a collection of input skills. Solutions to this task have mainly adopted graph operations and at least have the following limitations: (1) they are computationally demanding, as they require finding shortest paths on large collaboration networks; (2) they use various types of heuristics to reduce the exploration space over the collaboration network to become practically feasible; therefore, their results are not necessarily optimal; and (3) they are not well-suited for collaboration network structures given the sparsity of these networks. Our work proposes a variational Bayesian neural network architecture that learns representations for teams whose members have collaborated with each other in the past. The learned representations allow our proposed approach to mine teams that have a past collaborative history and collectively cover the requested desirable set of skills. Through our experiments, we demonstrate that our approach shows stronger performance compared to a range of strong team formation techniques from both quantitative and qualitative perspectives.

CCS Concepts: • **Information systems** → *Retrieval models and ranking*; **Expert search**; **Learning to rank**; • **Computing methodologies** → *Search methodologies*; **Learning latent representations;**

Additional Key Words and Phrases: Team formation, expert networks, task assignment, variational Bayesian neural network

ACM Reference format:
Radin Hamidi Rad, Hossein Fani, Ebrahim Bagheri, Mehdi Kargar, Divesh Srivastava, and Jaroslaw Szlichta. 2023. A Variational Neural Architecture for Skill-based Team Formation. *ACM Trans. Inf. Syst.* 42, 1, Article 7 (August 2023), 28 pages.
https://doi.org/10.1145/3589762

The implementation is available at: https://github.com/radinhamidi/A-Variational-Neural-Architecture-for-Skill-based-Team-Formation.

Authors' addresses: R. Hamidi Rad, E. Bagheri, and M. Kargar, Toronto Metropolitan University, Toronto, ON, Canada; emails: {radin, bagheri, kargar}@torontomu.ca; H. Fani, University of Windsor, Windsor, ON, Canada; email: hfani@uwindsor.ca; D. Srivastava, AT&T Chief Data Office, Bedminster, NJ, USA; email: divesh@research.att.com; J. Szlichta, York University, Toronto, ON, Canada; email: szlichta@yorku.ca.

1046-8188/2023/08-ART7 $15.00
https://doi.org/10.1145/3589762

1 INTRODUCTION

Complex projects in various application domains are led and performed by groups of experts, referred to as *teams*. This includes teams of experts in scientific research domains (e.g., to work on new technology or to conduct new research), engineering (e.g., to build new products), or the medical domain (e.g., to develop new cures or drugs). This is expected, as a single individual does not have the knowledge or expertise, nor the capacity to perform a complex project while delivering good results. Thus, forming a team of experts to perform a given project is an interesting and increasingly useful problem in many areas.

In a general setting, a project is essentially composed of a set of required tasks (e.g., a project needs expertise in data management and machine learning). In the basic form of the team formation problem, for a given set of required expertise, and a given pool of individuals, the task is concerned with finding a group of individuals that possess the required set of expertise. The pool of individuals might be provided by different online services, depending on the application's needs. This could be an employment-oriented service like LinkedIn for hiring employees, code hosting services like GitHub for hiring software engineers, and publication-hosting websites, such as DBLP and Google Scholar for hiring researchers. However, given a large pool of individuals, when potentially thousands of individuals possess each required expertise, selecting the right experts for any given task is far from trivial.

In general, the team formation problem can be considered as a variation of the set covering problem [25]. In the set covering problem, given a set of elements, namely, the universe and a collection of sets whose union equals the universe, the problem is to identify the smallest sub-collection of sets whose union equals the universe [9, 25]. Similarly, in the team formation problem, the set of all skills can form a universe and each expert can be considered as a set that has a subset of skills. Therefore, the team formation problem is to find the smallest sub-collection of experts whose union covers the target set of skills. Although using a solution for the set covering problem can form teams, it does not guarantee that the formed team is optimal. We will mathematically define an optimal team in Section 3. In the team formation problem, we look for an optimal team that in addition to covering skills can also benefit other aspects such as productivity and good communication among team members based on past collaboration.

The task of forming a team is not limited to only finding experts that possess a set of required skills. In real-world scenarios, there are additional important factors in the expert selection process. While in every real-world example there are many experts that have qualifications to be assigned to a job, their productivity and performance in a specific team remain in question. This is because team dynamics may be affected by many other factors such as interpersonal relationships. Thus, the team formation problem can face challenges regarding successful collaboration in many ways. For instance, experts may prefer to work with other experts in their company with whom they have successfully collaborated in the past. As a result, a successful team formation method would need to consider additional aspects in tandem with experts' skills. One way to define a criterion for selecting a group of individuals (i.e., a team) to successfully perform a new project is to make sure individual members of that team have past collaborations. This is because the success of a group project does not only depend on each individual's skills, but also on how effectively all members communicate and collaborate with each other. In general, successful collaboration in the past helps us find potential teams for future collaborations. We use past collaboration as a guide for finding the optimal team. Therefore, given the fact that the set covering problem cannot take past collaborations into account, it is not a suitable approach to optimally solve the team formation problem.

In prior work, to take past collaborations into account, first a network (i.e., graph) of experts is built [23, 32]. As shown in Figure 1, for forming the network, each individual is a node, and

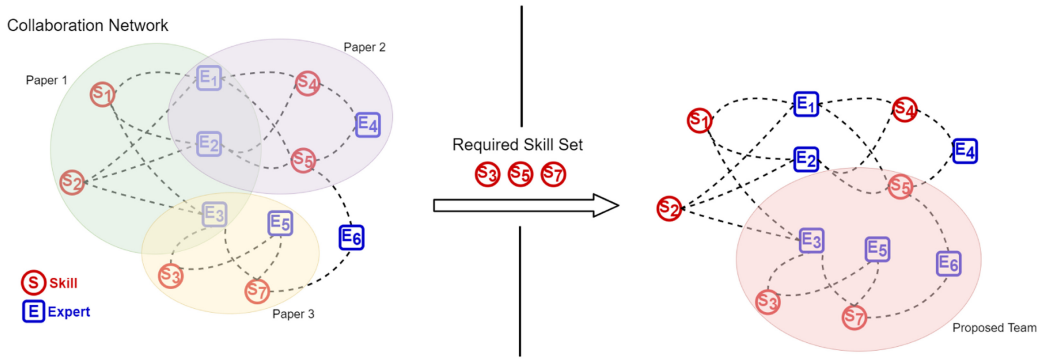

Fig. 1. A slice of a collaboration network. The expert and skill nodes are shown along with papers they have collaborated on in the past. Given a set of required skills, the Team Formation task is proposing a group of authors that collectively cover the skills.

past collaboration is modelled by connecting the associated nodes of individual members together to make an edge. Then, a graph search algorithm is employed to find a subgraph of the network, in which individual members collectively cover all required skills. Although different objective functions (such as the diameter of the subgraph) were proposed to choose the best subgraph, this problem can essentially be reduced to the classic *Group Steiner Tree* problem [32], hence, our proposed approach can potentially be used to solve the Group Steiner Tree problem, as well.

The focus of our work in this article is to find finding optimal teams by maximizing two main objectives: (1) coverage of the input skills of interest by the members of the team; and (2) effectiveness of past collaboration experience between the members of the team. To this end, similar to Sapienza et al. [52], we also propose to use a neural architecture for the team formation task; however, our approach is designed specifically to be cognizant of the collaboration network sparsity problem and its impact on overfitting. For this reason, we propose a novel variational Bayesian neural architecture. Different from existing work that considers the whole collaboration network as a search space, we adopt a *mapping function learning* strategy (as opposed to a search strategy) to map between expert and skill spaces. We show in our experiments that such a strategy is more effective and efficient than the existing state-of-the-art team formation methods.

This article extends and builds on our earlier work on the team formation [49] and offers the following contributions:

(1) We formally define the problem of learning to form a team of experts, as opposed to existing works that extract a subgraph from the expert network. We propose a *variational Bayesian neural* architecture[1] that learns the associations between experts and skills based on the composition of previously formed teams.
(2) We propose variations of our neural architecture and examine how these variations impact the outcome of the team formation task. We identify and discuss the structure of the optimal variation for this task.
(3) We compare our proposed model over real datasets (DBLP and Dota2) with strong baseline methods, including neural-based and collaborative filtering baselines, as well as graph-based methods. We show the efficiency and effectiveness of our approach in terms of standard metrics such as MAP, MRR, NDCG, F1, and recall as well as empirical training time.

[1]Note that the variational Bayesian neural architecture in this article is substantially expanded compared to the published short paper cited in Reference [49].

(4) We also evaluate our work based on three qualitative metrics in addition to the conventional quantitative metrics to evaluate the efficiency of the proposed approach from a qualitative/practical point of view. These metrics measure how productive and functional the proposed team can be in reality.

The structure of this article is as follows: Section 2 presents an extensive discussion of the related work and how we differ from them. Section 3 presents the problem statement and proposed method (including model architecture and training). Section 4 presents our extensive experiments. Section 5 concludes this article.

2 RELATED WORK

In any context where people are directly involved in the development of a final product, team formation plays a critical role. In the software industry, a software team performs a set of activities to deliver a high-quality software product and related artifacts (e.g., source code, models, test cases, and documentation) while satisfying a set of constraints, including cost, time, and scope, known as the triple constraint [11]. In **Concurrent Engineering (CE)**, which involves making decisions, multi-functional team formation has been extensively studied [60]. In crowdsourcing and open web-based collaboration like Wikipedia, it has been shown that those authors who collaboratively edit articles with each other are considered to be more efficient than those authors who are unrelated to each other [13]. Across available web services in distributed systems such as the cloud, matchmaking services for discovery and composition of services in open-systems have been also studied and can be considered as another related line of work in team formation where the members of the team are web services that need to effectively work together to accomplish a task [16, 35, 46, 58].

The earliest analytical models of the team formation problem primarily consisted of multiple objective functions that needed to be optimized via an **integer linear and/or nonlinear programming (IP)**, given constraints for human factors and non-human factors as well as scheduling preferences [4, 15, 19, 39, 40]. For instance, Baykasoglu et al. [4] proposed two fuzzy objective functions, namely, suitability, i.e., maximizing members' fit to the team based on their levels of expertise (skill), and team size, given hard-crisp constraints such as availability and salary of the team candidates (human factors) and schedule constraints for start and due dates (non-human factors). Such work, however, was premised on the mutually independent selection of team members among candidates and overlooked the organizational and/or social ties among individuals as well as past (un)successful collaborations. Therein, the optimization objectives were primarily driven by individuals' skills in isolation to find the best fit. A team is, however, inherently relational and is a property of the interaction among the team members and how effectively they can collaborate, which is often overlooked in the proposed mathematical optimization models.

Social relations can be modelled by integer programs by adding additional constraints to their objective functions such that they would consider experts' social ties such as the frequency of their previous interactions. Along this line, Wi et al. [55], introduced a non-linear optimization model based on a social network of candidates that represented interpersonal "familiarities" and candidates' skill sets to form teams. Co-authorship networks had already been shown to be effective by Cheatham et al. [10] on published works that capture previous successful collaborations among scientists. Dorn et al. [13] premise that an effective team needs a careful tradeoff between skill coverage and team collaboration effectiveness, especially in crowdsourcing environments. In these environments, experts routinely take up work assignments as they become available and therefore may not be able to join a team that is more suited to their skill sets at a given point in time. As such, when an expert who could be a part of an optimum team is not available, her social

network relations could be employed for recommending other candidates. For instance, in social trust networks, there are transitive relations among members and an unavailable candidate can be replaced by another 1-hop distant candidate that although not directly connected to the identified team, is still able to mediate second-hand knowledge among the unavailable candidate's neighbors. Dorn et al.'s work [13] is among the first to propose considering candidates' social neighborhood in their respective communities, atop skills and expertise of single members.

A step forward in this regard has been to employ **social network analysis (SNA)** to incorporate (i) social ties and (ii) interpersonal attributes such as communication, (iii) collaboration attributes such as the number of projects and the amount of time they worked together, and (iv) social attributes such as their level of friendship and the number of co-worker friends they have in common [20, 23, 32, 53]. For instance, coordination, communication, and cooperation can be analyzed using measures of SNA, such as density, degree centrality, and closeness centrality [44]. Further, social network analysis enables looking into the information flow in an organization. In this spirit, candidates (individuals) are often modelled in attributed weighted networks whose nodes are candidates with individual attributes (e.g., skills), and weighted links are established either explicitly or inferred based on interpersonal attributes between candidates. Gaston et al. [20] were among the first to study the effect of social network structure on the overall team formation performance. Given a group of agents, they form an attributed network whose nodes are agents (candidates) associated with a single skill as their attribute and edges are the organizational structure. Via empirical analysis on synthesis networks in simulation environments, Gaston et al. showed that the network structure has a notable impact on the number of possible optimum teams for a given set of tasks. Particularly, they showed scale-free networks whose degree distribution follows a power law are able to cover more optimum teams, hence, can be more efficient, for the set of tasks compared to those of types lattice and small-world. Gaston et al., however, have not proposed any computational models to maximize the efficacy of a given network or any optimization technique to find a valid team.

More recent computational models are based on the Steiner tree problem [31] in graphs where an optimal interconnection for a given set of nodes based on a predefined objective function is required. In this line, Lappas et al. [32] considered a co-authorship network to form a single team for performing a task. Like Gaston et al. [20], they define a team as a subgraph that minimizes the communication cost among the team members via some heuristics on the subgraph such as having the shortest diameter or being a **minimum spanning tree (MST)**. Unlike Gaston et al. [20] and Wi et al. [55], the proposed graph-based optimization method is limited to forming a single team at a time given a task and, hence, no race condition in team membership for performing multiple tasks is implied. Later, Sozio et al. [53] proposed a generalized umbrella definition for the class of team formation problems in graphs based on monotone optimization functions, so-called *cocktail party*: Given an undirected social graph of candidates G, a monotone optimization function f, and a set of constraints defined as monotone functions c_i, an "appropriate" team is an induced connected subgraph that maximizes f among all induced subgraphs of G while satisfies the constraints. Lappas et al.'s team formation, hence, is an instance of cocktail party assuming f is either minimum spanning tree or minimum diameter and constraints c_i are skill coverage of the incident nodes in the identified subgraphs.

Another instance of Sozio et al.'s cocktail party is proposed by Kargar et al. [23], who consider the monotone optimization function to be the sum of distances between the candidates in the induced connected subgraph that is to be minimized. Contrary to Lappas et al.'s MST and shortest diameter variations, Kargar et al. argue that the diameter and MST as communication cost functions are only able to address a limited aspect of a team's communication. They further propose that the sum of distances between experts might be a more effective measure of communication

effectiveness. Zihayat et al. [59] proposed a weighted attributed collaboration social network where edges, as well as nodes, are weighted. Like Lappas et al. and Sozio et al., weighted edges show the degree of successful collaboration in the past and hint at the communication cost that is to be minimized. Additionally, nodes (candidates) are weighted, e.g., h-index in author-network, which implies the prominence of the candidate and is supposed to be maximized. Therefore, an optimum team is a connected induced subgraph via a bi-factor optimization function that minimizes the communication cost while maximizing the candidates' weight within the subgraph. Keane et al. [26, 27] have extended Lappas et al.'s work by augmenting the explicit candidate collaboration network by potential links via link prediction algorithms followed by Lappas et al.'s heuristic method to identify minimum spanning subgraph as the optimum team. Via link prediction, the proposed method is able to form teams whose members might have not collaborated before, yet there is a high chance of successful collaboration in a future yet-to-be-formed team.

Algorithms on the formation of groups and communities in large-scale social networks could also be employed for the task of team formation including community detection [17, 18, 33], compact attributed group detection [29], and keyword search over attributed graph [7, 24]. For instance, given a graph G (e.g., the collaboration network), a set of query keywords (e.g., skills), and a range for the required number of nodes (e.g., candidates in a team) containing at least one keyword each, a compact attributed group is a connected subgraph of G that is composed of a set of nodes that are connected via their shortest path, and the size of the group is limited by lower and upper bounds. As seen, given skills, compact attributed graph detection can be employed to form a team whose size is limited within a range. Indeed, compact attributed group detection [29] is an instance of Sozio et al.'s cocktail party with size restriction [53].

There are several issues with the graph-based methods. The first major limitation of these methods relates to their scalability. Given these methods rely on graph operations that are primarily dependent on finding the shortest paths on the collaboration network, they are not efficient on real-world networks that consist of a large number of experts, skills, and past collaboration history. For example, the time complexity of the graph-based method proposed by Kargar et al. [24] is $O(N.t.|C_{max}|)$, where t is the number of query keywords, N is the number of nodes in the input graph (i.e., the graph among all experts built based on past collaboration), and $|C_{max}|$ is the maximum size of the content node sets for all input keywords. In this time complexity analysis, the authors assume that the shortest path can be computed in constant time using an advanced indexing method such as the 2-Hop Cover Labeling method [2]. Since there is no cap on the number of nodes containing the query keywords (i.e., $|C_{max}|$), in the worst-case scenario, this time complexity is quadratic in the number of nodes in the graph. Unlike graph-based methods, with complexity tied to the number of nodes (and edges) in the graph, the complexity of a neural approach is significantly lower when users aim to find an answer. This is because the complexity of the neural network depends on the number of computations within the hidden layers, which is significantly lower than the number of operations in graph-based methods.

There are several efforts [1, 27, 28, 45] and in particular the work by Khalil et al. [28] that propose models to solve the Group Steiner Tree problem. Khalil et al. have trained a graph neural network using reinforcement learning to incrementally construct a solution for NP-hard problems. This provides an opportunity for learning heuristic algorithms that exploit the structure of graphs to solve problems such as Group Steiner Tree.

Within the existing literature, the work by Sapienza et al. [52] is among the first to explore neural architectures for the team formation task. Despite its novelty in using an autoencoder architecture that leads to faster computation, it does face limitations such as being prone to overfitting [6], which leads to suboptimal performance primarily due to the sparse nature of collaboration network structures. The other work by Nikzad-Khasmakhi et al. [42] uses a neural architecture for expert

representation learning, which is used to retrieve experts by calculating similarity scores between required skills and experts. While recent neural team discovery techniques, such as [47], solely rely on learning a specific mapping function between the collaboration network node types (e.g., mapping from skill nodes to expert nodes), our work goes beyond such limited mappings utilizing a novel neural network architecture that learns the associations between experts and skills based on the composition of past collaborations.

Nonetheless, the above-proposed optimization models for the task of team formation were all computationally intractable and had to be followed by polynomial-time heuristic solutions such as multichoice [3] for subgraph identification with shortest diameter in Reference [32] or simulated annealing [4, 13], branch-and-cut [43], genetic algorithms [13, 55], greedy local search [39, 40], and balanced placement [15, 19] for those based on **integer programming (IP)**. Indeed, IP is NP-hard, and subgraph optimizations have been shown to be a reduced version of the Steiner-tree problem; an age-old NP-hard problem [25].

Unlike existing approaches, which adopt a search strategy over the whole collaboration network through either graph search methods or via IP mathematical optimization given selection and scheduling constraints, we take a statistical machine learning approach to learn a mapping function between skill and expert spaces.

3 PROPOSED METHOD

3.1 Problem Statement

Let $\mathcal{S} = \{s_i\}$ and $\mathcal{E} = \{e_j\}$ be the sets of skills and experts, respectively, $(\mathbf{s}, \mathbf{e})$ is a team of experts $\mathbf{e} \subseteq \mathcal{E}; \mathbf{e} \neq \emptyset$, which has been formed with respect to skill subset $\mathbf{s} \subseteq \mathcal{S}; \mathbf{s} \neq \emptyset$, and $U = \{(\mathbf{s}, \mathbf{e})_k\}$ indexes all teams. Our task is to learn $f : \mathcal{P}(\mathcal{S}) \rightarrow \mathcal{P}(\mathcal{E})$, a mapping function of parameters $\mathbf{\Theta}$ from skill powerset to experts powerset, such that $f(\mathbf{s}; \mathbf{\Theta}) = \mathbf{e}$. In other words, given a set of required skills $\mathbf{s}$ for a task, our goal is to assign an *optimal* team of experts $\mathbf{e}$ that are able to successfully accomplish the task. In Section 1, we have discussed that past collaborations can be used as a factor in forming teams. In this article, we define the optimal team as a team that was observed in the past. Let $\mathcal{T}$ be a subset of universe skill-expert sets that represent collaborations in the past. Then:

$$\mathcal{T} = \{(\mathbf{s}, \mathbf{e})_k\}, \mathcal{T} \subset U. \tag{1}$$

In the rest of the article, we learn a mapping function f based on the skills-experts sets from $\mathcal{T}$ and consider them as our reference dataset for the training, validation, and test purposes.

To address this problem, We propose a variational Bayesian neural network to estimate f and form an optimal team $(\mathbf{s}, \mathbf{e})$ of experts $\mathbf{e}$ for a required skill subset $\mathbf{s}$. Our work is motivated by the following considerations:

(1) Neural models enable us to go beyond the limited modeling capacity of graph-based models that heavily rely on searching the whole graph structure and allow us to selectively explore graph sub-spaces in a more efficient and targeted way. Despite long-standing promising performance in graph neural networks [41, 57], natural language understanding [12, 37], recommendation systems [34, 54], to name a few, neural networks have received less attention in the team formation literature. As such, we explore how neural networks are a faster and closer proxy for modelling teams.

(2) Training datasets in the team formation domain are quite imbalanced and sparse, that is, the majority of the skills and experts have only been involved in very few teams, which can cause overfitting towards the over-sampled skills or experts. Although overfitting can be addressed via introducing noise to the training samples $\mathcal{T}$ or to the predictions of the mapping function f by regularization via dropouts or optimization during training, the measure

of uncertainty in samples would not be embedded in the model architecture. We propose to employ a *variational Bayesian* neural network that incorporates uncertainty in the parameters of our mapping function Θ in the form of probability distributions (e.g., Gaussian). This allows for probabilistic weights as opposed to single real-valued weights. Our proposed Bayesian neural networks is, hence, robust to overfitting and can therefore effectively learn from sparse and highly skewed training samples.

In general, Bayesian neural networks are robust against overfitting [8, 22]. Moreover, in the team formation problem, sparsity is one of the main challenges and robustness against it is crucial. Sparsity in the team formation problem mainly exists because of the limited number of observed collaborations for each individual, which makes it hard for the model to learn effective collaborations. This is natural, because, in real-world scenarios, the number of collaborations for a small group of experts is not significant compared to the large number of experts in the collaboration network. Also, there is another challenge referred to as the long-tail problem, that is, few experts have the most collaboration (popular ones) but the majority have little. So, the canonical neural networks would overfit the popular ones, overlooking non-popular ones. Bayesian neural networks calculate the distribution of input features in their stochastic latent space [8]. As a result, Bayesian neural networks act as generative models that have good generalization and can perform better compared to discriminative models when fewer samples are available. Therefore, they can address sparsity in the collaboration network.

Moreover, Bayesian neural networks calculate the posterior inferences and bring uncertainty to predictions. In other words, the Bayesian neural network offers a solid approach for the quantification of uncertainty in deep models [22]. This is because, as mentioned earlier, their latent space is stochastic [8, 22] and learns collaborations by generalizing them using the calculated distribution of input features and achieves better generalization. As a consequence, the trained Bayesian neural network is robust to new cases, outliers, and fewer seen collaborations.

3.2 Model Architecture

We aim at estimating $f(\cdot, \Theta)$ using a multi-layer neural network with variational Bayesian layers. In Figure 2, we show an overview of our proposed variational Bayesian neural architecture. Given $(\mathbf{s}, \mathbf{e}) \in \mathcal{T}$, our model learns to map (transform) the input pretrained dense vector representation of the skill subset $v_{\mathbf{s}}$ and expert subset $v_{\mathbf{e}}$ to the occurrence vector representation of expert set $v_{\mathcal{E}}$ through two variational hidden layers.

The occurrence vector is a vector of unique skills or experts w.r.t. to its use case. Each element in this vector refers to a specific skill or expert. For instance, if a skill exists, then its value in the vector would be one, and if a skill does not exist, then it would be zero. This method for skill and expert representation results in sparse matrices, therefore, a dense vector representation is introduced to address this issue. In dense vector representations, the occurrence vectors are passed to a Word2Vec [36] model in the skip-gram setup to generate embedding vectors to represent skill and expert vectors.

The other novelty of our proposed approach is that we have designed the neural architecture in a way that it can benefit from team members' supervision during the skill to expert mapping process. This means that during training, skills and experts from a past collaboration are entered into two separate channels. We have used dense vectors of skills and experts as the input and represent them by $v_{\mathbf{s}}$ and $v_{\mathbf{e}}$, respectively. On the output side, experts are represented using an occurrence vector and are represented as $v_{\mathcal{E}}$. Considering the two separate channels, we can rewrite the mapping function as we estimate $f : \mathcal{P}(\mathcal{S}) \times \mathcal{P}(\mathcal{E}) \rightarrow \mathcal{P}(\mathcal{E})$ via training instances of $(\mathbf{s}, \mathbf{e}) \in \mathcal{T}$ in the training step.

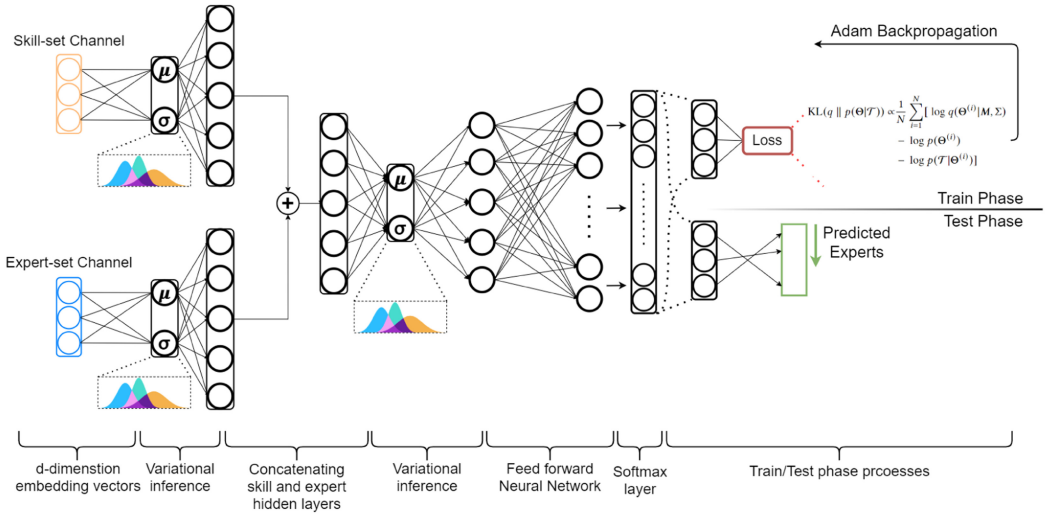

Fig. 2. The overview of our proposed approach.

During the prediction step, the dense expert vector $v_{\mathbf{e}}$ will be replaced with an all-zeros vector, as it is unknown. However, the difference in input content between the training and test phases does not imply a difference in architecture. While the architecture remains the same, it is possible to forgo including experts in the input layer and only provide skills by passing zeros as the representation of experts. This provides the additional benefit of enabling the model to be used for predicting teams that have a set of skills and are seeded by one or more experts. In other words, this will be useful for cases when we are interested in forming teams around one or more initial experts with a given set of skills. As such, one can input the expert channel with a specific expert to force the architecture to form teams more related to that person or add multiple experts so the model tries to complete the input team with additional experts. Therefore, at the prediction step, we employ $f : \mathcal{P}(\mathcal{S}) \rightarrow \mathcal{P}(\mathcal{E})$ to select an accurate subset of experts from all available experts in $\mathcal{E}$ as the appropriate members for the given skill subset $\mathbf{s} \subseteq \mathcal{S}$ whose team $(\mathbf{s}, \mathbf{e})$ has not been seen before. Formally,

$$\mathbf{h}_{\mathbf{s}}^0 = v_{\mathbf{s}}; \mathbf{h}_{\mathbf{e}}^0 = v_{\mathbf{e}}; \tag{2}$$

$$\mathbf{h}_{\mathbf{s}}^1 = \sigma(\mathbf{\Theta}_{\mathbf{s}}^1 \mathbf{h}_{\mathbf{s}}^0 + \mathbf{b}_{\mathbf{s}}^1); \mathbf{h}_{\mathbf{e}}^1 = \sigma(\mathbf{\Theta}_{\mathbf{e}}^1 \mathbf{h}_{\mathbf{e}}^0 + \mathbf{b}_{\mathbf{e}}^1), \tag{3}$$

$$\mathbf{h}^2 = \sigma(\mathbf{\Theta}^2 [\mathbf{h}_{\mathbf{s}}^1 \frown \mathbf{h}_{\mathbf{e}}^1] + \mathbf{b}^2), \tag{4}$$

$$v_{\mathcal{E}} = \sigma(\mathbf{\Theta}^3 \mathbf{h}^2 + \mathbf{b}^3), \tag{5}$$

where σ is a sigmoid activation function, $v_{\mathbf{s}}$ and $v_{\mathbf{e}}$ are the dense vector representation of size d for $\mathbf{s}$ and $\mathbf{e}$, respectively, $v_{\mathcal{E}}$ is the output layer of size $|\mathcal{E}|$ whose elements are in $\mathcal{R}^{[0,1]}$, and $\mathbf{\Theta} = \mathbf{\Theta}^{\cdot} \cup \mathbf{b}^{\cdot}$ are parameters of our mapping function $f(\mathbf{s}; \mathbf{\Theta}) = \mathbf{e}; \forall (\mathbf{s}, \mathbf{e}) \in \mathcal{T}$.

Herein and contrary to *non*-variational neural models where the parameters are real values (point estimate), our model's parameters $\theta \in \mathbf{\Theta}$ are random variables (probabilistic parameters) each of which follows a probabilistic distribution $p(\theta)$ and bring uncertainty in parameters. Our task is to estimate the probability distributions for each parameter. In our proposed model, we assume that the parameters are independently distributed and each is drawn from Gaussian distribution q with its own mean and variance, i.e., $\theta \sim \mathcal{N}(\mu, \sigma^2)$ or $q(\theta) = \mathcal{N}(\mu, \sigma^2)$. In other words, our proposed neural model for estimating the mapping function f is indeed characterized by pairs of

(μ, σ^2) for each parameter. We refer to all means and variances by $\boldsymbol{M}$ and Σ. We estimate the parameters' means and variances by our proposed variational inference via minimizing variational free energy, which we explain in the next section.

3.3 Variational Inference

Given $\mathcal{T}$ as the set of teams whose elements $(\mathbf{s}, \mathbf{e})$ consist of an input skill subset $\mathbf{s}$ and a target expert subset $\mathbf{e}$, which are assumed to be drawn independently from a joint distribution $p(\mathbf{s}, \mathbf{e})$, We aim at optimizing the **maximum a posteriori (MAP)** of Θ in $f(\cdot, \Theta)$, i.e., $p(\Theta|\mathcal{T})$ where f is a multi-layer neural network with variational Bayesian layers and Θ are probabilistic parameters, or weights. By Bayes theorem,

$$p(\Theta|\mathcal{T}) \propto p(\mathcal{T}|\Theta)p(\Theta) \quad \text{where} \quad p(\mathcal{T}|\Theta) = \prod_{(\mathbf{s},\mathbf{e})\in\mathcal{T}} p(\mathbf{e}|\mathbf{s}, \Theta) \tag{6}$$

and $p(\Theta)$ is the prior join probability of weights that is unknown. Maximizing $p(\mathcal{T}|\Theta)p(\Theta)$ gives the maximum a posteriori estimate of Θ. The true prior probability of weights $p(\Theta)$, however, cannot be calculated analytically or efficiently sampled, and as such, we approximate it by a more tractable distribution $q(\Theta|\boldsymbol{M}, \Sigma)$ with multivariate diagonal Gaussian distribution $\mathcal{N}(\boldsymbol{M}, \Sigma^2)$. The elements of Σ are a diagonal covariance matrix, which means that weights Θ are assumed to be uncorrelated (independently distributed).

To estimate the true posterior $p(\Theta)$ by $q(\Theta|\boldsymbol{M}, \Sigma)$, we minimize the Kullback-Leibler divergence between q and p with regard to the Gaussian means and variances as suggested by Graves [21]:

$$\mathrm{KL}(q \parallel p(\Theta|\mathcal{T})) = \int q(\Theta|\boldsymbol{M}, \Sigma) \log \left[\frac{q}{p(\Theta|\mathcal{T})}\right] \mathrm{d}\Theta \tag{7}$$

$$= \mathbb{E}_q \log \left[\frac{q}{p(\Theta|\mathcal{T})}\right] \tag{8}$$

$$\text{where} \quad \mathrm{KL}(q \parallel p) \geq 0. \tag{9}$$

In the context of Bayesian learning, $\mathrm{KL}(q \parallel p)$ is referred to as the information to be lost if our estimate q is used to approximate the true p, or, conversely, the information to be gained by revising our model's belief from the prior probability distribution q to the true probability distribution q. By applying Bayes theorem to Equation (9), we obtain:

$$\mathrm{KL}(q \parallel p(\Theta|\mathcal{T})) = \mathbb{E}_q \log \left[\frac{q}{p(\mathcal{T}|\Theta)p(\Theta)} p(\mathcal{T})\right] \tag{10}$$

$$= \mathbb{E}_q \log \left[\frac{q}{p(\Theta)p(\mathcal{T}|\Theta)} p(\mathcal{T})\right] \tag{11}$$

$$= \mathbb{E}_q \log \left[\frac{q}{p(\Theta)}\right] - \mathbb{E}_q \log p(\mathcal{T}|\Theta) + \mathbb{E}_q \log p(\mathcal{T}). \tag{12}$$

Using the fact that the log marginal likelihood $\log p(\mathcal{T})$ is not dependent on Θ:

$$\mathrm{KL}(q \parallel p(\Theta|\mathcal{T})) = \mathbb{E}_q \log[\frac{q}{p(\Theta)}] - \mathbb{E}_q \log p(\mathcal{T}|\Theta) + \log p(\mathcal{T}) \tag{13}$$

$$= \underbrace{\mathrm{KL}(q \parallel p(\Theta)) - \mathbb{E}_q \log p(\mathcal{T}|\Theta)}_{\textit{variational free energy}} + \log p(\mathcal{T}). \tag{14}$$

To minimize $\mathrm{KL}(q \parallel p(\boldsymbol{\theta}|\mathcal{T}))$, we need to minimize the first two terms in Equation (14), known as variational free energy, given the fact that the log marginal likelihood $\log p(\mathcal{T})$ does not depend on $\boldsymbol{M}$ and Σ. The first term in variational free energy is the Kullback-Leibler divergence between

our estimate distribution $q(\Theta|\boldsymbol{M}, \Sigma)$ and the prior $p(\Theta)$ and is called the *complexity cost,* and the second term is the expected value of the loglikelihood with respect to the q and is called the *likelihood cost.* By expanding the complexity cost, our KL minimization can be written as:

$$\mathrm{KL}(q \parallel p(\Theta|\mathcal{T})) \propto \mathbb{E}_q \log q - \mathbb{E}_q \log p(\Theta) - \mathbb{E}_q \log p(\mathcal{T}|\Theta). \tag{15}$$

As seen, all three terms in Equation (15) are expectations based on our estimate distribution $q(\Theta|\boldsymbol{M}, \Sigma)$. Therefore, our minimization can be approximately calculated by drawing random samples Θ from our estimate q.

$$\begin{aligned}\mathrm{KL}(q \parallel p(\Theta|\mathcal{T})) \propto \frac{1}{N} \sum_{i=1}^{N} [& \log q(\Theta^{(i)}|\boldsymbol{M}, \Sigma) \\ & - \log p(\Theta^{(i)}) \\ & - \log p(\mathcal{T}|\Theta^{(i)})].\end{aligned} \tag{16}$$

3.4 Model Training

Similar to *non*-variational feed-forward neural networks, our training iterations consist of forward passes to calculate the KL minimization function approximately in Equation (17) (a.k.a. approximate cost or loss function) and backward passes. During a forward pass and via a stochastic sampling step, a sample set of parameters Θ is drawn from the variational posterior distribution $q(\Theta|\boldsymbol{M}, \Sigma) = \mathcal{N}(\boldsymbol{M}, \Sigma^2)$. The first two terms of Equation (17) can be calculated in parallel to the forward pass, since they are based on the expectation of the parameters with respect to q and do not depend on the input. The last term, however, depends on the given set of teams $\mathcal{T}$ and is evaluated at the end of the forward pass. During the backward passes, we back-propagate the gradients of $\boldsymbol{M}$ and Σ and update the means and variances of parameters. Recall that the parameters of our neural model Θ are random variables drawn from the variational posterior distribution $q(\Theta|\boldsymbol{M}, \Sigma) = \mathcal{N}(\boldsymbol{M}, \Sigma^2)$. Therefore, our minimization in Equation (17) is, in fact, a function of means $\boldsymbol{M}$ and variances Σ.

The procedure of the training phase is presented in detail in Algorithm 1. First, we compute the embedding vectors of the skill and expert sets in Lines 2–4. These embedding vectors will then be passed as inputs to the neural network in Lines 7 and 8. As shown in Figure 2, the neural network performs variational inference on each of the two parallel inputs (Lines 10–11). After computing μ and σ for each of the inputs, they will be passed to a hidden layer (Lines 13–14). The hidden layers are then concatenated and form the second hidden layer (Line 16). The second-stage variational inference will use the concatenated layer from the previous step to generate the next hidden layer result (Line 18). After computing μ and σ, we will pass the results to a multi-layer perceptron to learn the final expert probabilities (Lines 20–22). The loss functions used for model training is based on Equation (17).

3.5 Model Prediction

Once our variational Bayesian neural model has learned its best variational posterior distribution:

$$\hat{q}(\Theta|\boldsymbol{M}, \Sigma) = \mathcal{N}(\hat{\boldsymbol{M}}, \hat{\Sigma}^2), \tag{17}$$

which maximizes MAP of Θ in $p(\Theta|\mathcal{T})$ by KL minimization in Equation (9) via approximate minimization of the variational free energy in Equation (17), given an unseen $(\mathbf{s}, \mathbf{e})$, we input the dense vector representation of $\mathbf{s}$, i.e., $v_{\mathbf{s}}$, to our model in Equation (2). Through a forward pass, we randomly draw the values of our neural model's parameters in Equation (3) from the learned $\mathcal{N}(\hat{\boldsymbol{M}}, \hat{\Sigma}^2)$. Our predicted team members $\hat{\mathbf{e}} \subseteq \mathcal{E}$ are those experts who have the top-k highest values in the output layer $v_{\mathcal{E}}$ in Equation (5). As seen, the prediction is a stochastic process that brings

ALGORITHM 1: Training the model

Input: $\mathcal{T} = \{(\mathbf{s}, \mathbf{e})\}$.
1 **begin**
2 **foreach** (s_i, e_i) *in* $(\mathbf{s}, \mathbf{e})$ **do**
3 $f_{Embedding} \leftarrow$ Train embedding based on skip-gram model$(s \longleftrightarrow e)$
4 **end**
5 **foreach** (s_i, e_i) *in* $(\mathbf{s}, \mathbf{e})$ **do**
6 Fetch the embedding vectors:
7 $\bar{s_i} \leftarrow f_{Embedding}(s_i)$
8 $\bar{e_i} \leftarrow f_{Embedding}(e_i)$
9 Sub-sampling and calculating distribution parameters μ and σ:
10 $h^0_{s_i} \leftarrow \mu_{\bar{s_i}}, \sigma_{\bar{s_i}}$
11 $h^0_{e_i} \leftarrow \mu_{\bar{e_i}}, \sigma_{\bar{e_i}}$
12 A hidden layer after variational layer:
13 $h^1_{s_i} = f_{sigmoid}(\Theta^1_{s_i} h^0_{s_i} + b^1_{s_i})$
14 $h^1_{e_i} = f_{sigmoid}(\Theta^1_{e_i} h^0_{e_i} + b^1_{e_i})$
15 Concatenating the layers from two channels:
16 $h^2_i \leftarrow Concatenate(h^1_{s_i}, h^1_{e_i})$
17 Sub-sampling and calculating distribution parameters μ and σ:
18 $h^3_i \leftarrow \mu_{h^2_i}, \sigma_{h^2_i}$
19 A feed-forward neural network:
20 $h^4_i = f_{sigmoid}(\Theta^3_i h^3_i + b^3_i)$
21 $y = f_{softmax}(h^4_i)$
22 Minimize $\mathcal{L} \propto \frac{1}{N} \sum_{i=1}^{N} [\log q(\mathbf{\Theta}^{(i)} | \boldsymbol{M}, \Sigma) - \log p(\mathbf{\Theta}^{(i)}) - \log p(\mathcal{T} | \mathbf{\Theta}^{(i)})]$
23 **end**
24 **end**

uncertainties in team formation that arise from the uncertainty in parameters, also called epistemic uncertainty, as opposed to aleatoric uncertainty due to noise in our dataset $\mathcal{T}$. While epistemic uncertainty is higher when no or little teams of experts are available for skill subsets and can be reduced by more samples, it best suits our purpose for team formation where the distribution of experts in skill subsets are sparse.

The detailed procedure of prediction phase is shown in Algorithm 2. To derive our candidate experts to form a team, we first need to generate embedding vectors of a given skill set (Line 4). Take note that, since we are in the testing phase, we have no clue of the target experts, therefore, we eliminate the expert embedding vector at input. We calculate μ and σ of the given input in Line 6 and pass it through a hidden layer (Line 7). Since the neural network architecture used to have two input channels during training, namely, skill and expert channels, we need to replace the expert input channel result with a zero vector with identical dimensions instead. In the next step, we concatenate the two vectors in Line 8 and perform the second variational inference on them (Line 9). The results will then be passed to a multi-layer perceptron (Lines 10–11). Line 11 in Algorithm 2 will be the final step of our neural network. The size of this layer is equal to the total number of experts. Since values of these numbers are output of a softmax function, we can consider each value to be a probability score for their referred expert. At the last step, in Lines 13–15, we will sort these probabilities in descending order and pick the top k experts as the candidate experts that can be members of our proposed team.

4 EXPERIMENTS

4.1 Datasets and Setup

To perform experiments to evaluate the performance of our team formation approach, we require datasets that have ground truth teams included in them already without requiring manual labeling of the data. Earlier work such as the work by Zihayat et al. [59], Khan et al. [29], and Lapas et al. [32], have suggested that datasets that represent collaborative work by a set of experts would qualify for

ALGORITHM 2: Prediction procedure

Input: Skill-set: $\mathcal{S} = \{s_i\}$
Parameters : Top k results(k)
Output: Predicted expert-set: $\mathcal{E}' = \{e'_i\}$

begin
  foreach s_i *in* $\mathcal{S}$ **do**
    Fetch the embedding vectors:
    $\bar{s_i} \leftarrow f_{Embedding}(s_i)$
    Running neural network in forward direction:
    $h^0_{s_i} \leftarrow \mu_{\bar{s_i}}, \sigma_{\bar{s_i}}$
    $h^1_{s_i} = f_{sigmoid}(\Theta^1_{s_i} h^0_{s_i} + b^1_{s_i})$
    $h^2_i \leftarrow Concatenate(h^1_{s_i}, zeros(shape : h^1_{e_i}))$
    $h^3_i \leftarrow \mu_{h^2_i}, \sigma_{h^2_i}$
    $h^4_i = f_{sigmoid}(\Theta^3_i h^3_i + b^3_i)$
    $y = f_{softmax}(h^4_i)$
    Sort the output based on their score:
    $y_{sorted} = Sort_{Dsc}(y)$
    Top k predicted experts:
    $\mathcal{E}' \leftarrow y_{sorted}[1:k]$
  end
end

this purpose. For instance, the DBLP dataset includes a comprehensive set of publications within the field of Computer Science and consists of a group of authors that have collaboratively co-authored a scientific paper together. The set of authors on a paper can be seen as a team. Similarly, the Dota2 dataset consists of details for each of the matches played. Each match consists of two groups of five players that compete against the opponent group. At the beginning of the match, players choose their hero and statistics related to each match are recorded at the end. The winner group can be considered as a team and match configuration, e.g., selected heroes will be skill sets.

In our experiments, we adopt the DBLP[2] and Dota2[3] datasets. For DBLP dataset, the authors of each paper are collectively considered to represent a team; hence, each author is considered to be an expert. The skills associated with each author are the keywords that appear in the publications of that author based on the method suggested in References [23, 32, 48, 50, 51]. Briefly, the keywords are extracted from each author's paper titles by first applying pre-processing, i.e., performing stemming and eliminating stop-words. The remaining $\{1, 2, 3\}$-grams are sorted based on their TF-IDF scores and the top terms are reviewed by the authors of this article to check their quality and avoid any duplication. Then, the top 2,000 are retained as the set of skills for the experiments. We also monitored the distribution of terms per article using the histogram chart in Figure 3 to make sure that there is a reasonable number of terms per article. In the end, the dataset consists of 33,002 teams 2,000 skills (keywords) and 2, 470 experts (authors). The distribution of skills over the team sizes is shown in Figure 3.

For the Dota2 dataset, we considered each player who was part of a winning team to be an expert and the players who are in the same group in a match to form a team. We formed the skill set of each match using its attributes. These attributes include (1) those related to the in-game setup, i.e., selected heroes, tower status, and barracks status; and (2) those related to the configuration of the game itself, i.e., the game version, server region, and server cluster. In the original dataset, plenty of players did not sign in with their accounts while playing the game. We have dropped matches with anonymous players and then filtered out those matches with players that appeared only once. In summary, 6,390 teams (group of players) and 3,005 skills and 2,727 experts (players)

[2] https://www.aminer.org/citation.
[3] https://www.kaggle.com/datasets/devinanzelmo/dota-2-matches.

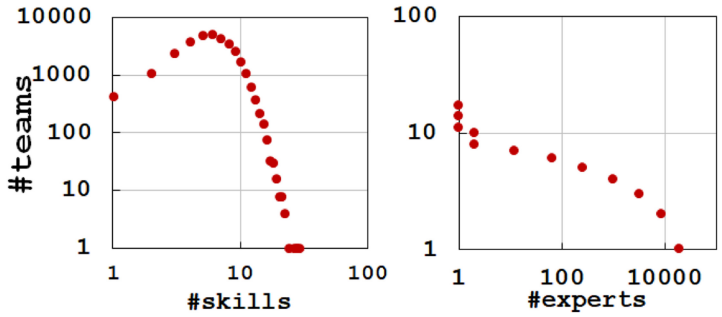

Fig. 3. DBLP dataset. Left: The histogram chart represents the distribution of teams over the size of skills they own. The majority of the teams have around 10 skills. Right: The histogram chart showing the distribution of experts over the number of times they appeared in a team.

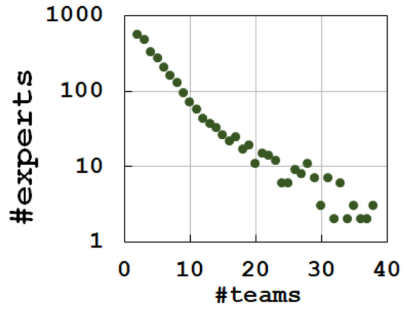

Fig. 4. Dota2 dataset. The histogram chart represents the distribution of experts (players) in regard to the number of times each of them participated in a team (match). Unlike the DBLP dataset, there is no histogram for the skillset size of the teams; that is because skills are reflecting game setup, which is fixed to 3,005.

are included in this dataset. A histogram chart to show the distribution of players in regard to the number of times each of them participated in a match can be seen in Figure 4.

Finally, we note that in our work, we adopt two representation types, namely, (1) occurrence vector representation (o); and (2) pre-trained dense vector representation (p) to model skills and experts. For the sake of readability, we include subscripts o and p to each representation to denote whether an occurrence vector or dense vector representation has been used, respectively.

All code and data related to our work are publicly available for further reproducibility and expansion studies.[4]

4.2 Baselines

To benchmark our work against strong state-of-the-art baselines, we adopt three classes of baseline methods. The first type of methods are those that form teams by performing graph heuristics to find a subset of a collaboration network that would represent an effective team. We adopt two **graph-based** methods:

- The work by Kargar et al. [24] can be considered the strongest baseline in graph-based methods, which proposes to model team formation as a variation of subgraph identification through keyword search on graphs.

[4]https://github.com/radinhamidi/A-Variational-Neural-Architecture-for-Skill-based-Team-Formation.

- The method proposed by Lappas et al. [32] views a team as a minimum-diameter subtree. The main objective of this graph-based method is to form a team by maximizing the collaboration level of the team members.

In addition, it is possible to define the problem of team formation as a collaborative filtering task where team members are recommended based on a set of required skills. We adopt the following **collaborative filtering** baselines:

- The RRN method proposed by Wu et al. [56] employs factorization along with considering future behavioral trajectories by using an LSTM-based autoregressive model.
- We also include the well-known SVD++ method [30] that employs matrix factorization and also takes implicit interactions as well as user and item biases into account.
- Also, Du et al. [14] in their recent paper named GERF, proposed a new learning to rank method based on **Bayesian Group Ranking (BGR)** that utilize Bayesian inference to optimize the weights of the model. They have crafted feature vectors as the input for their learning-to-rank algorithm.

Finally, we also include three neural team formation methods in our baselines. To the best of our knowledge, these are the only other **neural-based methods** for the problem of team formation:

- The first work is by Sapienza et al. [52], which is in essence an autoencoder that learns the adjacency matrix representing the experts that are linked to each other on the collaboration network.
- The other work is by Nikzad-Khasmakhi et al. [42], which we refer to as ExEm in this article. This work employs a neural architecture for expert representation learning, which is used to retrieve experts by calculating similarity scores between required skills and experts.
- Last work is by Khalil et al. [28], which we refer to as S2V (structure to vector) in this article. In S2V, the authors trained a graph neural network model using reinforcement learning to generate embedding vectors to solve heuristic algorithms. After the training phase, the model will be able to generate representation vectors suitable for solving the Group Steiner Tree problem. We have used these representation vectors to solve the team formation problem.

We note that, for all methods, we used the default hyperparameter settings proposed by the authors.

4.3 Evaluation Strategy

We evaluate the performance of our proposed work from the two perspectives of *efficacy* and *efficiency*.

4.3.1 Efficacy. For the sake of measuring efficacy, we use a 10-fold cross-validation strategy where the mapping function $f : \mathcal{P}(\mathcal{S}) \rightarrow \mathcal{P}(\mathcal{E})$ is trained on the teams that are observed in the training folds, which is then used to predict the teams that are placed in the test fold. We compare the predicted team $f(s, \theta) = e$ with the observed team e in (s, e) from the test fold.

We report our findings from two different perspectives, namely, *ranking* and *matching* perspectives. In the ranking approach, we measure the quality of the ranked list of experts retrieved for a given set of skills. The most effective team formation method would be one that ranks the correct members of the team at the top of the retrieved ranked list. As such, we measure the efficacy of the ranking of the experts using four ranking metrics, namely, *average recall @k*, ***mean average precision (MAP), mean reciprocal rank (MRR)***, and ***normalized discounted cumulative gain (NDCG)***. The *average recall @k* that is calculated is based on an exact match. This means for a given set of skills, the model is successful only if it proposes the exact team of experts as expected.

In addition, we measure the efficacy of the proposed method from the perspective of matching. Here, the objective is to measure the quality characteristics of the identified teams according to three metrics, namely, **Skill Coverage (sk), Team Formation Success (tfs)**, and **Communication Cost (cc)**.

Skill Coverage (sk): This metric measures to what extent the recommended team completely covers the set of skills that are specified. This metric is similar to the recall metric with one difference: Instead of seeking the exact same set of experts as expected, skill coverage will be looking to see whether all the required set of skills are covered regardless of the chosen experts. In other words, this removes the exact match requirement of the recall metric. The purpose of this metric is to reward those models that form teams that do not consist of the exact set of ground truth experts but still consist of experts that have all the required expertise.

Team Formation Success (tfs): This metric measures the percentage of the desired skills has been covered by the proposed team. In other words, team formation success shows the extent to which the formed teams were able to cover the required set of skills. The score equals the coverage ratio. In contrast to the skill coverage metric (sk), models still get a score if they cover the required skills partially and not all of them.

Communication Cost (cc): The purpose of this metric is to show whether the formed teams have effective past collaboration history or not. The *cc* metric measures the average shortest path distance between the team member pairs. This metric will be minimized when the expected team is predicted. This is because the expected team members are those experts who have worked together directly in the past and are hence connected to each other on the collaboration network.

4.3.2 Efficiency. Further, we measure the efficiency of our proposed method compared to the baselines by measuring scalability. We report scalability as a function of the execution time (training and inference) of each method when executed on the same dataset and using the same computational resources. The computational resources used in our experiments included a 12 CPU core-server with 64 GB memory and a GPU unit of 3,584 cores and 12 GB memory.

4.4 Ablation Study

Given our proposed architecture can accommodate different types of skill and expert representations. Here, we explore the impact of different skill and expert representations on the performance of our proposed architecture. To represent experts and skills, we adopt two representation types: (1) occurrence vector representation (o); and (2) pre-trained dense vector representation (p). The adoption of these two representation types leads to four variations when training $f : \mathcal{P}(\mathcal{S}) \rightarrow \mathcal{P}(\mathcal{E})$ where S and E can be represented through o and p representation types. For the sake of readability, we include a subscript for S and E to denote their representation type. For example, $S_p \rightarrow E_o$ indicates that a pre-trained dense skill representation is used in the input and an occurrence expert vector is used at the output.

We additionally include four further variations of our architecture by not only including the skill vector representations in the input but also incorporating expert vector representations in the input as well. In other words, the architecture will learn to effectively map a set of skills and their associated experts to the expert space, i.e., $f : \mathcal{P}(\mathcal{S}) \times \mathcal{P}(\mathcal{E}) \rightarrow \mathcal{P}(\mathcal{E})$. As an example, $S_pE_p \rightarrow E_o$ denotes the concatenated pre-trained dense skill and representation vectors as the input and the occurrence expert vector as the output.

The results of our ablation study are reported in Figures 5, 6, 7, and 8 for the DBLP and Dota2 datasets, respectively. We make three main observations from the ablation study:

(1) The variation of our model that learns to jointly map skills and experts in the input to the experts in the output, i.e., $f : \mathcal{P}(\mathcal{S}) \times \mathcal{P}(\mathcal{E}) \rightarrow \mathcal{P}(\mathcal{E})$ shows a significantly better performance

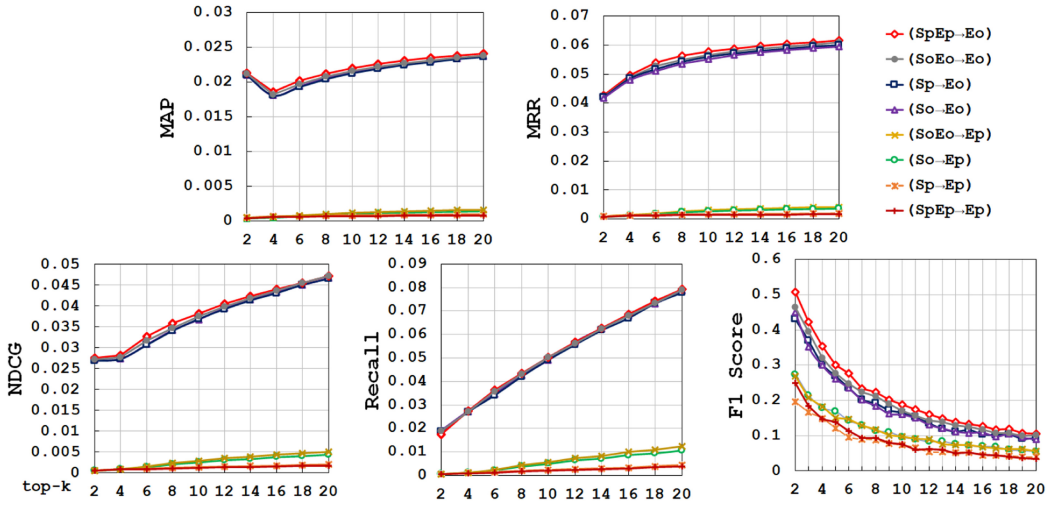

Fig. 5. The performance of the variations of our approach on the DBLP dataset.

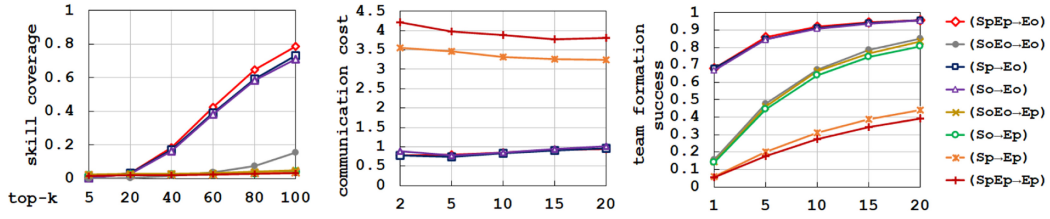

Fig. 6. Qualitative performance of the variations of our approach on the DBLP dataset.

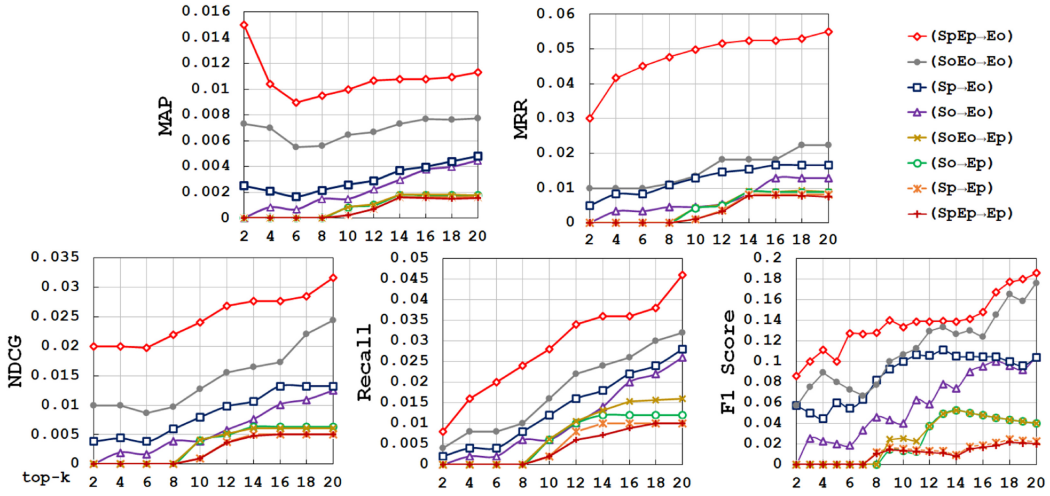

Fig. 7. The performance of the variations of our approach on the Dota2 dataset.

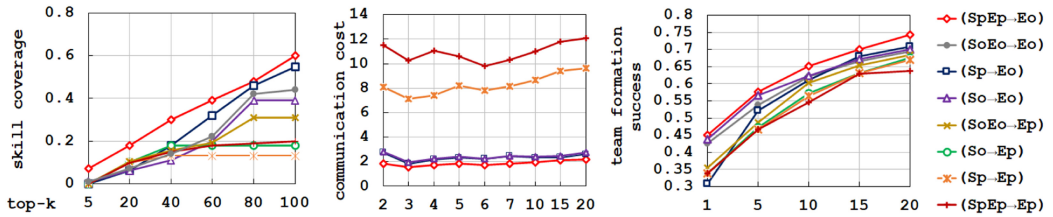

Fig. 8. Qualitative performance of the variations of our approach on the Dota2 dataset.

compared to cases when skills are mapped to the experts, i.e., $f : \mathcal{P}(\mathcal{S}) \rightarrow \mathcal{P}(\mathcal{E})$. This can be due to the fact that when including both skills and experts in the input, the mapping function is able to more effectively find the relationship between the elements of the expert and skill spaces given they are jointly fed to the model in the input layer.

(2) When considering the expert representations, the variations that consider E_p in the output perform consistently worse than the variation that uses the occurrence expert vector representation at the output (i.e., E_o); that is because by using a pre-trained dense representation format, the objective of the task changes slightly. In such a case, the task changes from the multi-class classification problem to the regression problem. One drawback of using dense representation at the output is that we need to take a further step and find similar teams based on their distance to the predicted representation. We believe that two main factors can cause inferior performance: (a) In a regression task, the model needs to predict a dense representation vector and hence a slight change in the values may point to a completely different team; (b) since we need to find similar teams based on the distance between their vector representation, there would be a chance that multiple teams have very close similarities. But because only the most similar team will be chosen, the rest would not be considered. In contrast, in the occurrence vector approach, the confidence of membership for each expert is calculated individually and there is a chance for partial coverage.

(3) It can be seen that some variations are performing closely to each other, e.g., $S_pE_p \rightarrow E_o$ and $S_p \rightarrow E_o$. We have conducted significance tests on all the variations to ensure that they are statistically significant. For the statistical significance test, we perform a paired t-test with a confidence interval of 95%.

(4) Finally, we find that from among the eight variations of our approach, the $S_pE_p \rightarrow E_o$ shows the overall better performance over all four evaluation metrics and on both DBLP and Dota2 datasets. This is consistent with the previous two findings, as it is a variation that jointly considers both skills and experts in the input (i.e., $\mathcal{P}(\mathcal{S}; \mathcal{E})$) and also benefits from the occurrence expert vector representation at the output. For the sake of comparison with the other baselines and in the remainder of the article, we adopt the $S_pE_p \rightarrow E_o$ variation to represent our proposed work.

4.5 Comparison with Baselines

4.5.1 Ranking Metrics. We first compare the performance of our proposed method with the baselines based on the ranking metrics. The performance of the different baselines on ranking metrics are reported in Figures 9 and 10 for the DBLP and the Dota2 datasets, respectively. Given the fact that none of the baseline methods (including our method) is able to determine the correct number of team members automatically, and the team size needs to be provided as input, the x-axis of each diagram shows performance of the different methods for varying team sizes. Based on these figures, our main observation is that on both the DBLP and the Dota2 datasets and over

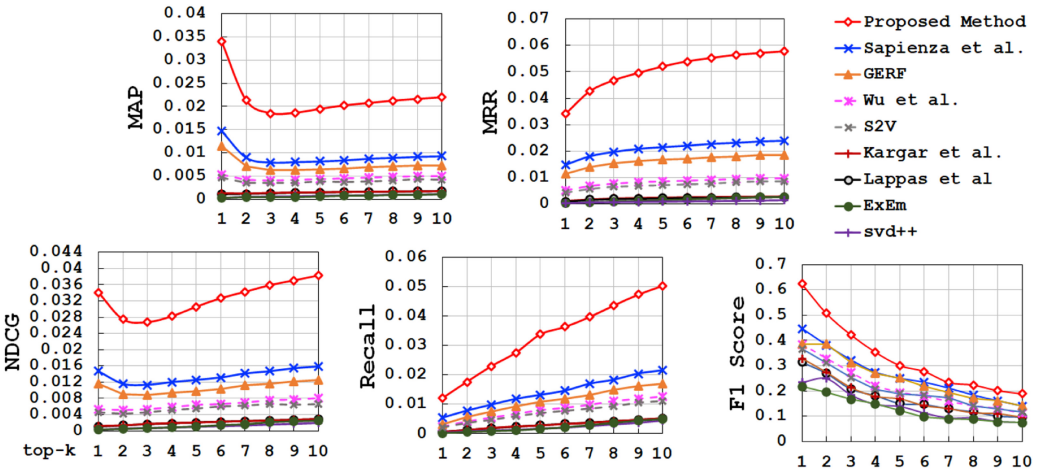

Fig. 9. The performance of our proposed model against the baselines for the DBLP dataset on ranking metrics.

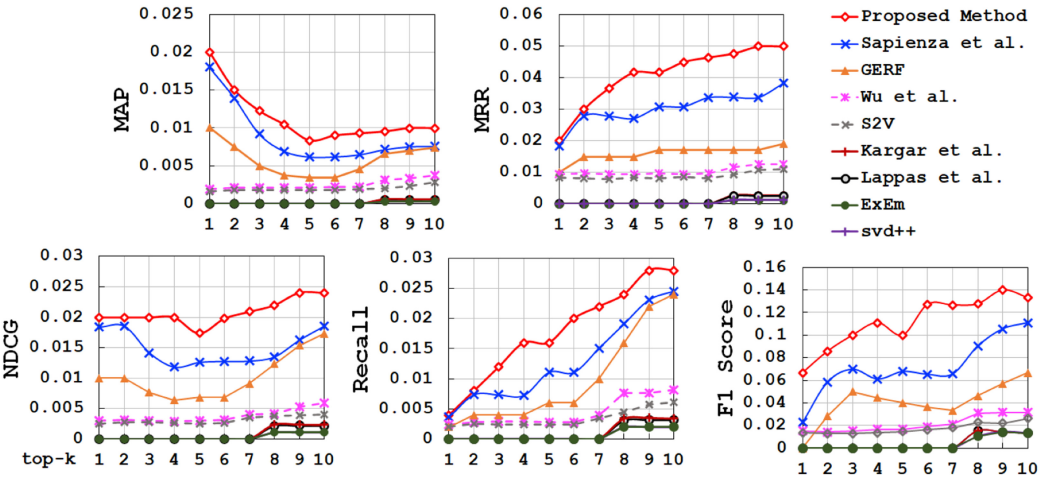

Fig. 10. The performance of our proposed model against the baselines for the Dota2 dataset on ranking metrics.

all four ranking metrics, our proposed method shows consistent better performance compared to the baseline methods. The performance improvement shown by our proposed method when considering a team size of 10 is at least 2.2× the best baseline on the DBLP dataset and at least 1.3× the best baseline on the Dota2 dataset.

Our second observation pertains to the performance of the other baselines. We observe that the best baseline method is the work proposed by Sapienza et al. [52], which uses a neural autoencoder architecture for team formation. Similarly the closely competing baselines for the second-best performance are the work by Wu et al. [56], which adopts a long short-term memory autoregressive model to make team member recommendations and GERF, which is based on Bayesian group ranking. As such, we find that methods based on neural architectures outperform those that are primarily based on graph-based heuristics. When observing the performance of the state-of-the-art

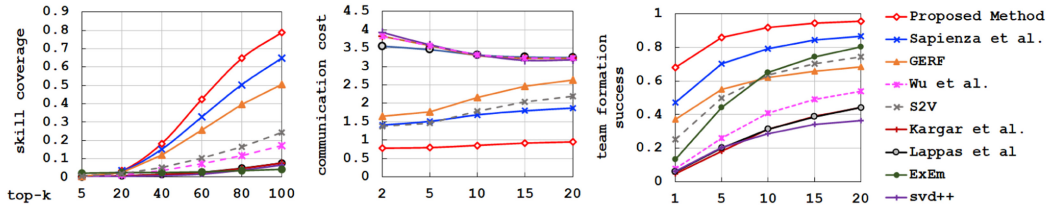

Fig. 11. Qualitative measures of performance on the DBLP dataset.

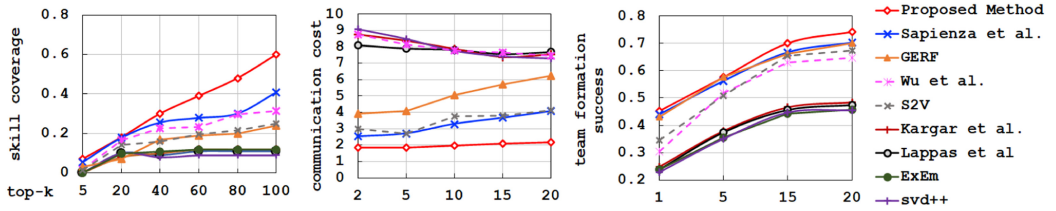

Fig. 12. Qualitative measures of performance on the Dota2 dataset.

graph-based team formation methods, such as the work by kargar et al. [24], we find that such methods are not able to provide competitive performance to not only our proposed method but also to the other baselines. The main reason for this is that graph-based methods need to repeatedly compute computationally expensive graph metrics such as shortest paths or cliques in the collaboration network and hence are practically infeasible to perform in large graphs. Hence, they resort to heuristic techniques that systematically consider subsets of the graph to reduce computational complexity. As such, this results in sub-optimal performance by these methods in practice.

4.5.2 Qualitative Metrics. In addition to ranking metrics, we compare the performance of our proposed approach to the baselines from a qualitative perspective as well. We adopt the three qualitative measures introduced earlier for this purpose, namely, team formation success, communication cost, skill coverage. We note that a higher value on team formation success and skill coverage is desirable, while a lower value is desired for communication cost. The results of the comparison on the DBLP and Dota2 datasets are shown in Figures 11 and 12, respectively. We observe that our approach is able to show better performance compared to all baselines on the three qualitative measures on both datasets. More concretely, we make the following observations:

- The first qualitative metric, i.e., skill coverage, considers whether the formed team includes experts that have all the required skills regardless of whether these experts were a part of the original team of the test set or not. We found out that by increasing the k in the top-k, our proposed method skill coverage increases with a faster trend in comparison to the baselines and reaches a higher coverage in general.
- The communication cost qualitative metric measures, regardless of the suitability of the proposed team, whether the members of the proposed team have effective past collaborations. As seen in the figure, our proposed approach has the lowest communication cost compared to the other baselines showing a desirable past communication history among the members of the proposed teams. Also, it can be seen that by increasing the k in top-k, the cost remains relatively stable for the proposed method, however, for most of the baselines, it increases. That means, for the proposed method, the ranked list of experts consists of those with past collaborations and by moving down the list, proposed experts still have past collaborations. In contrast, baselines do not follow this trend and the communication cost increases

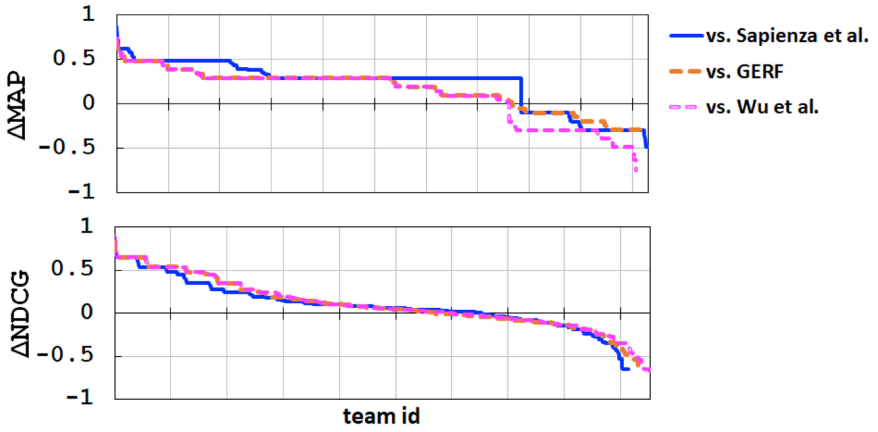

Fig. 13. Comparative analysis of performance on a team-basis on the DBLP dataset. Teams with an equivalent performance by our approach and the baseline have not been included ($\Delta = 0$).

gradually. It is noted that by increasing the k in top-k, other graph-based approaches i.e. Kargar et al. [24], Lapas et al. [32] and Wu et al. [56] remain stable and their communication cost drops slightly.

- From the perspective of the team formation success metric, the x-axis of the figures shows the number of additional candidates that need to be added for the formed team to have the same set of skills that the actual team has in the test set. As seen in both figures, our method is not only able to show a high value for the team formation success measure but also reaches faster. That means the proposed method needs a fewer number of team members (lower k in top-k) to be able to reach higher team formation success values.

On the DBLP and Dota2 datasets, our proposed approach shows superior performance over all baselines. We note that the proposed method constantly improved performance with growth in team size. This indicates that in our approach, additional experts in different depths of top-k contribute to the different skills of the team and do not necessarily have overlapping skills, whereas in some other baselines the experts that are retrieved in the top-k have the same set of skills and therefore skill coverage and team formation success scores do not change as the size of k increases.

4.5.3 Performance Robustness. It is important to show whether the observed improvements, as reported in the previous sections, are robust across the whole collection of teams and their associated skills. For this purpose, we compare the performance of our proposed method with the performance of the three strongest baselines, i.e., Sapienza et al. [52], Wu et al. [56], and GERF [14], on a per query basis. Given the performance of the four ranking metrics were consistent on both datasets, and for the sake of space, we report this comparative analysis on NDCG and MAP in Figures 13 and 14 for the DBLP and Dota2 datasets, respectively. In these figures, each point on the x-axis shows an instance of a team, and the y-axis shows the percentage of the difference between the MAP or NDCG of our proposed approach compared to that of the baseline. These figures show that the improvements reported in Section 4.5.1 are observable consistently on a large number of queries. More concretely, when considering the MAP metric and the DBLP dataset, our proposed method improves the average precision of 2,355 teams over Sapienza et al. while only hurting the performance of 740 teams (a similar performance on the other two baselines as well). This shows that the performance of our proposed approach is robust across a wide range of teams, consistently leading to performance improvements on a larger number of teams than those whose performances have been negatively impacted.

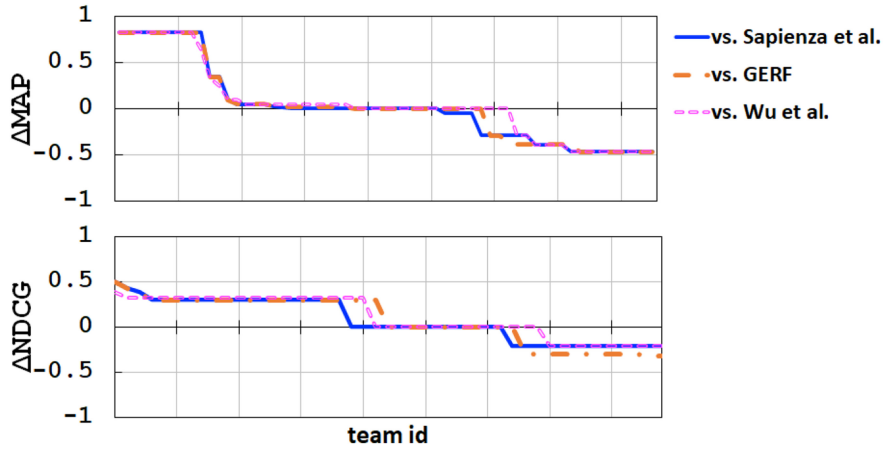

Fig. 14. Comparative analysis of performance on a team-basis on the Dota2 dataset. Teams with an equivalent performance by our approach and the baseline have not been included ($\Delta = 0$).

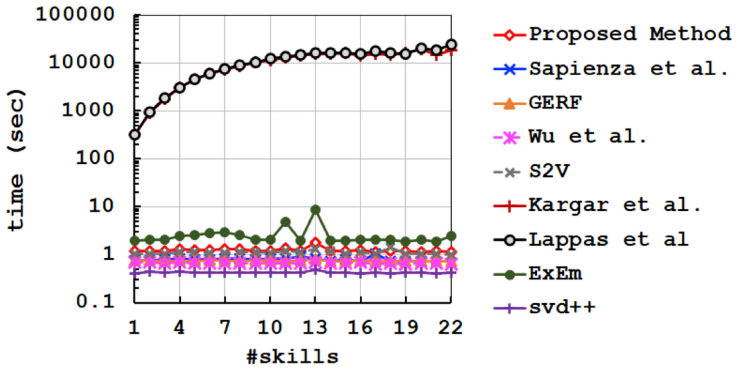

Fig. 15. The inference time on the DBLP dataset.

We also note that while the method by Sapienza et al. shows an overall better performance compared to the methods proposed by Wu et al. and GERF when comparing their performance at a per-team level, we find that the number of teams that are improved by Wu et al. and GERF is larger. This indicates that while Sapienza et al. is able to help a lower number of teams, the degree of efficacy of this method on those teams is higher. In contrast, Wu et al. and GERF show efficacy on a larger number of teams, but the improvements are smaller on those teams compared to the other baseline. However, our proposed method reports improvements over the three of these strong baselines in terms of the number of teams that have been effectively formed and the degree of improvement over the efficacy metrics.

4.5.4 Execution Efficiency. In addition to the efficacy of our proposed approach, we evaluate our work from the perspective of execution efficiency both at training time and at inference time. From the perspective of inference, there are two parameters that impact the performance of our model and the baselines, namely, the number of input skills and the size of the generated team. We report the comparative efficiency of our proposed approach compared to the baselines in Figure 15 for the DBLP dataset. In this figure, the x-axis denotes the number of input skills and the y-axis is the time taken to generate a team (inference time) in seconds. We make an observation

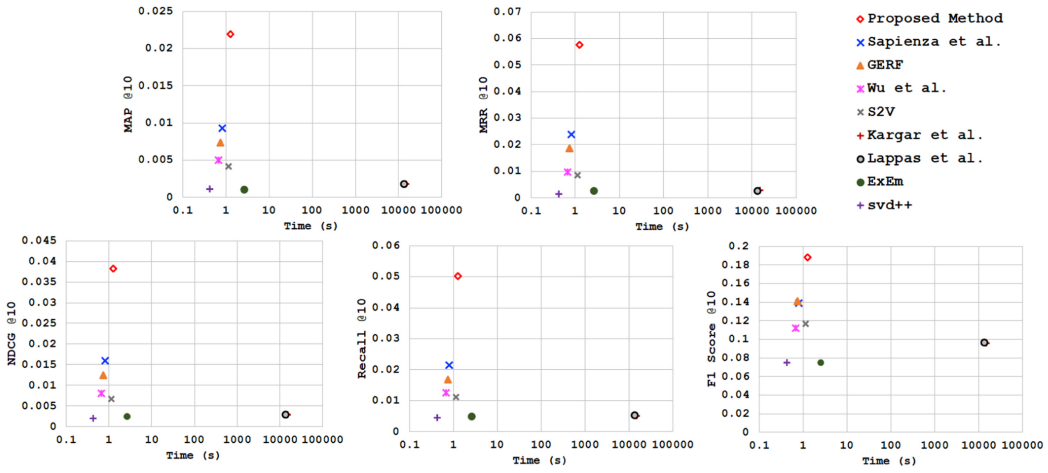

Fig. 16. The tradeoff between inference time and efficacy on the DBLP dataset.

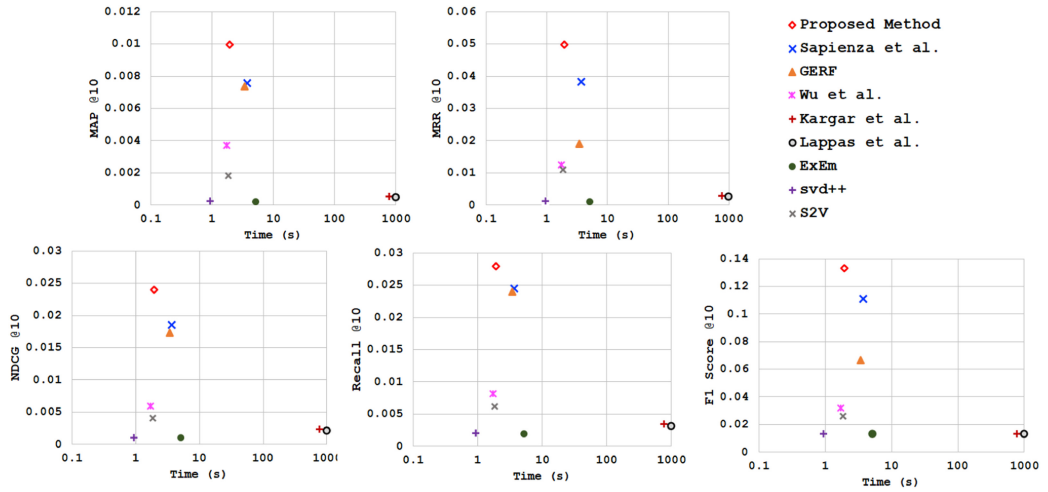

Fig. 17. The tradeoff between inference time and efficacy on the Dota2 dataset.

that the inference time of graph-based methods, namely, Kargar et al. [24] and Lapas et al. [32], is significantly larger (by orders of magnitude) than the other baselines in the collaborative filtering and neural categories. There are two reasons for this. First, graph-based methods do not have a training phase and therefore all operations of the methods are performed at the inference (team formation) time. Second, these methods rely on some variation of graph traversal and/or graph metric computation that are very time-consuming in practice. As such, the execution efficiency of graph-based methods is quite low. In contrast, the methods that consist of an initial training phase have faster inference (team formation) speeds. As shown in the figure, while our method shows competitive performance on inference time, it does not show faster inference compared to the other non-graph-based baselines. The most efficient baseline method is the work by Koren et al., which infers (forms) teams in ~0.42 second, on average.

It is important to contextualize this finding by showing the tradeoff between team formation (inference) time and the efficacy of the formed teams. To this end, Figures 16 and 17 show the

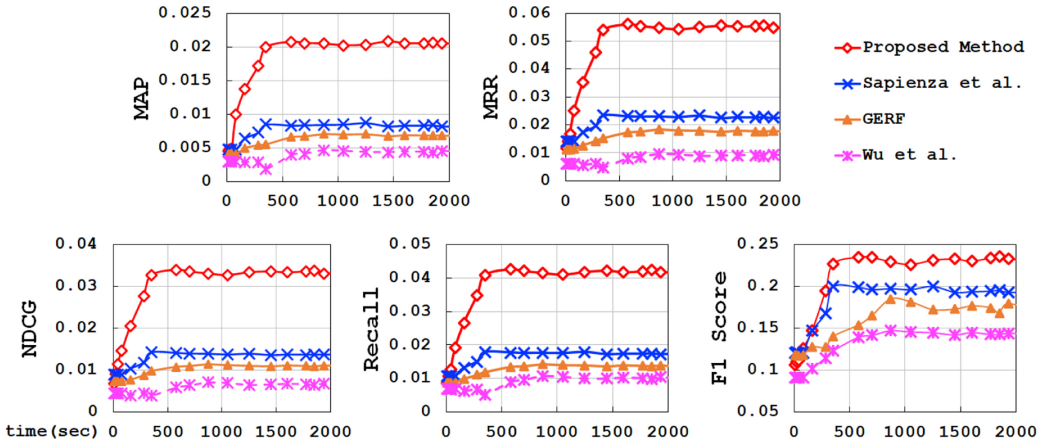

Fig. 18. Impact of training time on method efficacy on the DBLP dataset.

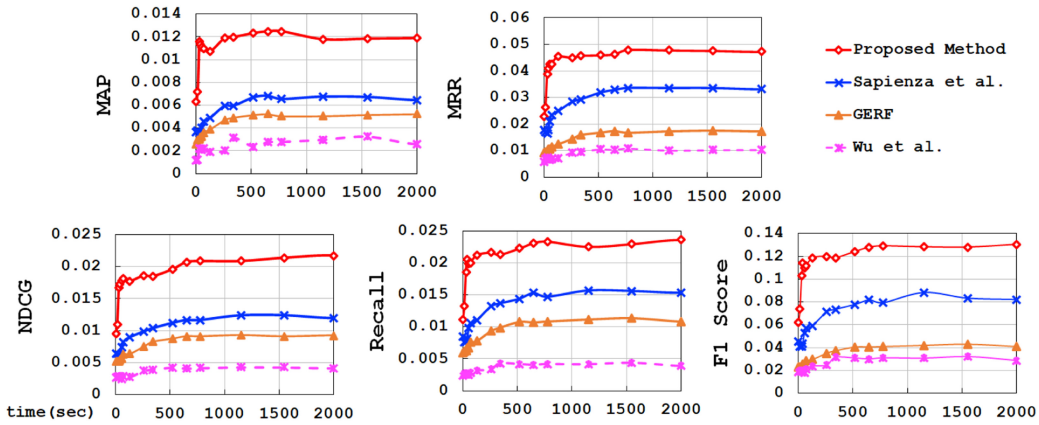

Fig. 19. Impact of training time on method efficacy on the Dota2 dataset.

tradeoff between inference time and efficacy metrics for the DBLP and Dota2 datasets, respectively. The figures show that regardless of the metric or the dataset, while the method by Koren et al. shows fast inference time, the quality of the teams generated by this method is quite poor. In these two figures, an ideal team formation method would be one that shows a fast inference time and high performance on the efficacy metrics. Based on the tradeoff between inference time and efficacy, a desirable team formation method would be one that is located in the top-leftmost corner of the figure. We observe that in both DBLP and Dota2 datasets, our method is placed on the top-left corner and shows the most desirable tradeoff between efficacy and inference time. Methods such as Sapienza et al. and GERF also have low inference times but they are not competitive to our proposed method on the efficacy metrics and as such are located lower compared to our method on the y-axis.

We also compare the performance of our proposed approach from the perspective of training time and inference's impact on efficacy. To this end, we report the efficacy of our trained model compared to the three best baselines in Figures 18 and 19. The efficacy metrics are computed on the test set after allowing the method to be trained for the amount of time shown on the x-axis. The

figures show that the four methods require a comparable amount of time to reach stability, while our proposed approach shows significantly better efficacy when trained for the same amount of time.

5 CONCLUDING REMARKS

In this article, we have modelled the problem of team formation through a stacked variational Bayesian neural network architecture. The advantage of our proposed approach is that it is light-weight, preserves the semantics of team structure including team members' past collaboration history, and captures the relationship between experts and their set of skills. To the best of our knowledge, our work is among the first to address the team formation problem through learning from past collaborations in the collaboration network. Prior approaches often rely on learning a task-specific mapping between the skill and expert spaces, which, due to the sparsity of the collaboration network, are not as efficient as our proposed approach or utilize conventional graph search that yields sub-optimal solutions. In this work, when the same skills are represented using different keywords, we consider them as different skills. This can be extended by using word analogy when building the skill set. One possible method is to adapt the prior work by Bergamaschi et al. [5] to identify semantically similar keywords based on ontologies, such as WordNet [38]. We plan to incorporate this in our future work. In our experiments, we compared our proposed approach with the state-of-the-art from three perspectives, namely, information retrieval, qualitative, and scalability. We find that our approach shows superior performance compared to the baselines on all ranking and quality metrics. It also provides a faster and more efficient training process.

REFERENCES

[1] Abu Reyan Ahmed, Md Asadullah Turja, Faryad Darabi Sahneh, Mithun Ghosh, Keaton Hamm, and Stephen G. Kobourov. 2021. Computing Steiner trees using graph neural networks. *CoRR* abs/2108.08368 (2021).

[2] Takuya Akiba, Yoichi Iwata, and Yuichi Yoshida. 2013. Fast exact shortest-path distance queries on large networks by pruned landmark labeling. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. ACM, 349–360. DOI: https://doi.org/10.1145/2463676.2465315

[3] Esther M. Arkin and Refael Hassin. 2000. Minimum-diameter covering problems. *Networks* 36, 3 (2000), 147–155. DOI: https://doi.org/10.1002/1097-0037(200010)36:3<147::AID-NET1>3.0.CO;2-M

[4] Adil Baykasoglu, Türkay Dereli, and Sena Das. 2007. Project team selection using fuzzy optimization approach. *Cybern. Syst.* 38, 2 (2007), 155–185. DOI: https://doi.org/10.1080/01969720601139041

[5] Sonia Bergamaschi, Elton Domnori, Francesco Guerra, Raquel Trillo Lado, and Yannis Velegrakis. 2011. Keyword search over relational databases: A metadata approach. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. ACM, 565–576. DOI: https://doi.org/10.1145/1989323.1989383

[6] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning (ICML'15)*. JMLR.org, 1613–1622.

[7] Spencer Bryson, Heidar Davoudi, Lukasz Golab, Mehdi Kargar, Yuliya Lytvyn, Piotr Mierzejewski, Jaroslaw Szlichta, and Morteza Zihayat. 2020. Robust keyword search in large attributed graphs. *Inf. Retr. J.* 23, 5 (2020), 502–524. DOI: https://doi.org/10.1007/s10791-020-09379-9

[8] Alexander Camuto, Matthew Willetts, Stephen J. Roberts, Chris C. Holmes, and Tom Rainforth. 2021. Towards a theoretical understanding of the robustness of variational autoencoders. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 130)*. PMLR, 3565–3573. Retrieved from http://proceedings.mlr.press/v130/camuto21a.html.

[9] Alberto Caprara, Paolo Toth, and Matteo Fischetti. 2000. Algorithms for the set covering problem. *Ann. Oper. Res.* 98, 1 (2000), 353–371.

[10] Michelle Cheatham and Kevin Cleereman. 2006. Application of social network analysis to collaborative team formation. In *Proceedings of the International Symposium on Collaborative Technologies and Systems*. IEEE Computer Society, 306–311. DOI: https://doi.org/10.1109/CTS.2006.18

[11] Alexandre Costa, Felipe Barbosa Araújo Ramos, Mirko Perkusich, Emanuel Dantas, Ednaldo Dilorenzo, Ferdinandy Chagas, André Meireles, Danyllo Albuquerque, Luiz Silva, Hyggo O. Almeida, and Angelo Perkusich. 2020. Team

formation in software engineering: A systematic mapping study. *IEEE Access* 8 (2020), 145687–145712. DOI: https://doi.org/10.1109/ACCESS.2020.3015017

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 4171–4186. DOI: https://doi.org/10.18653/v1/n19-1423

[13] Christoph Dorn, Florian Skopik, Daniel Schall, and Schahram Dustdar. 2011. Interaction mining and skill-dependent recommendations for multi-objective team composition. *Data Knowl. Eng.* 70, 10 (2011), 866–891. DOI: https://doi.org/10.1016/j.datak.2011.06.004

[14] Yulu Du, Xiangwu Meng, Yujie Zhang, and Pengtao Lv. 2020. GERF: A group event recommendation framework based on learning-to-rank. *IEEE Trans. Knowl. Data Eng.* 32, 4 (2020), 674–687. DOI: https://doi.org/10.1109/TKDE.2019.2893361

[15] Edmund H. Durfee, James C. Boerkoel Jr., and Jason Sleight. 2014. Using hybrid scheduling for the semi-autonomous formation of expert teams. *Fut. Gen. Comput. Syst.* 31 (2014), 200–212. DOI: https://doi.org/10.1016/j.future.2013.04.008

[16] Schahram Dustdar and Wolfgang Schreiner. 2005. A survey on web services composition. *Int. J. Web Grid Serv.* 1, 1 (2005), 1–30. DOI: https://doi.org/10.1504/IJWGS.2005.007545

[17] Yixiang Fang, Reynold Cheng, Siqiang Luo, and Jiafeng Hu. 2016. Effective community search for large attributed graphs. *Proc. VLDB Endow.* 9, 12 (2016), 1233–1244. DOI: https://doi.org/10.14778/2994509.2994538

[18] Hossein Fani, Eric Jiang, Ebrahim Bagheri, Feras N. Al-Obeidat, Weichang Du, and Mehdi Kargar. 2020. User community detection via embedding of social network structure and temporal content. *Inf. Process. Manag.* 57, 2 (2020), 102056. DOI: https://doi.org/10.1016/j.ipm.2019.102056

[19] Erin Fitzpatrick and Ronald G. Askin. 2005. Forming effective worker teams with multi-functional skill requirements. *Comput. Industr. Eng.* 48, 3 (2005), 593–608. DOI: https://doi.org/10.1016/j.cie.2004.12.014

[20] Matthew E. Gaston, John Simmons, and Marie desJardins. 2004. Adapting network structure for efficient team formation. In *Artificial Multiagent Learning, Papers from the 2004 AAAI Fall Symposium. Arlington, VA, USA, October 22-24, 2004*, Vol. FS-04-02. AAAI Press, 1–8. Retrieved from https://www.aaai.org/Library/Symposia/Fall/2004/fs04-02-001.php.

[21] Alex Graves. 2011. Practical variational inference for neural networks. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems*. 2348–2356. Retrieved from https://proceedings.neurips.cc/paper/2011/hash/7eb3c8be3d411e8ebfab08eba5f49632-Abstract.html.

[22] Laurent Valentin Jospin, Hamid Laga, Farid Boussaïd, Wray L. Buntine, and Mohammed Bennamoun. 2022. Hands-on bayesian neural networks—A tutorial for deep learning users. *IEEE Comput. Intell. Mag.* 17, 2 (2022), 29–48. DOI: https://doi.org/10.1109/MCI.2022.3155327

[23] Mehdi Kargar and Aijun An. 2011. Discovering top-k teams of experts with/without a leader in social networks. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management*. ACM, 985–994. DOI: https://doi.org/10.1145/2063576.2063718

[24] Mehdi Kargar, Lukasz Golab, Divesh Srivastava, Jaroslaw Szlichta, and Morteza Zihayat. 2022. Effective keyword search over weighted graphs. *IEEE Trans. Knowl. Data Eng.* 34, 2 (2022), 601–616. DOI: https://doi.org/10.1109/TKDE.2020.2985376

[25] Richard M. Karp. 1972. Reducibility among combinatorial problems. In *Proceedings of a Symposium on the Complexity of Computer Computations*. Plenum Press, New York, 85–103. DOI: https://doi.org/10.1007/978-1-4684-2001-2_9

[26] Peter Keane, Faisal Ghaffar, and David Malone. 2019. Using machine learning to predict links and improve Steiner tree solutions to team formation problems. In *Complex Networks and Their Applications VIII - Volume 2 Proceedings of the Eighth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2019, Lisbon, Portugal, December 10-12, 2019 (Studies in Computational Intelligence, Vol. 882)*, Hocine Cherifi, Sabrina Gaito, José Fernendo Mendes, Esteban Moro, and Luis Mateus Rocha (Eds.). Springer, 995–1006. DOI: https://doi.org/10.1007/978-3-030-36683-4_79

[27] Peter Keane, Faisal Ghaffar, and David Malone. 2020. Using machine learning to predict links and improve Steiner tree solutions to team formation problems—A cross company study. *Appl. Netw. Sci.* 5, 1 (2020), 57. DOI: https://doi.org/10.1007/s41109-020-00306-x

[28] Elias B. Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. 2017. Learning combinatorial optimization algorithms over graphs. In *Proceedings of the Annual Conference on Neural Information Processing Systems*. 6348–6358. Retrieved from https://proceedings.neurips.cc/paper/2017/hash/d9896106ca98d3d05b8cbdf4fd8b13a1-Abstract.html.

[29] Abeer Khan, Lukasz Golab, Mehdi Kargar, Jaroslaw Szlichta, and Morteza Zihayat. 2020. Compact group discovery in attributed graphs and social networks. *Inf. Process. Manag.* 57, 2 (2020), 102054. DOI: https://doi.org/10.1016/j.ipm.2019.102054

[30] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 447–456. DOI: https://doi.org/10.1145/1557019.1557072

[31] Bernhard H. Korte and Jens Vygen. 2011. *Combinatorial Optimization*. Vol. 1. Springer.

[32] Theodoros Lappas, Kun Liu, and Evimaria Terzi. 2009. Finding a team of experts in social networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 467–476. DOI: https://doi.org/10.1145/1557019.1557074

[33] Xiao Li, Chenna Sun, and Muhammad Azam Zia. 2020. Social influence based community detection in event-based social networks. *Inf. Process. Manag.* 57, 6 (2020), 102353. DOI: https://doi.org/10.1016/j.ipm.2020.102353

[34] Guoqiong Liao, Xiaobin Deng, Changxuan Wan, and Xiping Liu. 2022. Group event recommendation based on graph multi-head attention network combining explicit and implicit information. *Inf. Process. Manag.* 59, 2 (2022), 102797. DOI: https://doi.org/10.1016/j.ipm.2021.102797

[35] Zakaria Maamar, Djamal Benslimane, Philippe Thiran, Chirine Ghedira, Schahram Dustdar, and Sattanathan Subramanian. 2007. Towards a context-based multi-type policy approach for web services composition. *Data Knowl. Eng.* 62, 2 (2007), 327–351. DOI: https://doi.org/10.1016/j.datak.2006.08.007

[36] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations*. Retrieved from http://arxiv.org/abs/1301.3781.

[37] Tomás Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association*. ISCA, 1045–1048. Retrieved from http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html.

[38] George A. Miller. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

[39] Maryam Mirzaei, Jörg Sander, and Eleni Stroulia. 2019. Multi-aspect review-team assignment using latent research areas. *Inf. Process. Manag.* 56, 3 (2019), 858–878. DOI: https://doi.org/10.1016/j.ipm.2019.01.007

[40] Mahmood Neshati, Hamid Beigy, and Djoerd Hiemstra. 2014. Expert group formation using facility location analysis. *Inf. Process. Manag.* 50, 2 (2014), 361–383. DOI: https://doi.org/10.1016/j.ipm.2013.10.001

[41] Hoang Nguyen, Radin Hamidi Rad, and Ebrahim Bagheri. 2022. PyDHNet: A Python library for dynamic heterogeneous network representation learning and evaluation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, 4936–4940. DOI: https://doi.org/10.1145/3511808.3557181

[42] Narjes Nikzad-Khasmakhi, Mohammad Ali Balafar, Mohammad-Reza Feizi-Derakhshi, and Cina Motamed. 2021. ExEm: Expert embedding using dominating set theory with deep learning approaches. *Expert Syst. Appl.* 177 (2021), 114913. DOI: https://doi.org/10.1016/j.eswa.2021.114913

[43] Manfred Padberg and Giovanni Rinaldi. 1991. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Rev.* 33, 1 (1991), 60–100. DOI: https://doi.org/10.1137/1033004

[44] Carla Sofia Pereira and António Lucas Soares. 2007. Improving the quality of collaboration requirements for information management through social networks analysis. *Int. J. Inf. Manag.* 27, 2 (2007), 86–103. DOI: https://doi.org/10.1016/j.ijinfomgt.2006.10.003

[45] Chotipat Pornavalai, Norio Shiratori, and Goutam Chakraborty. 1996. Neural network for optimal {steiner} tree computation. *Neural Process. Lett.* 3, 3 (1996), 139–149. DOI: https://doi.org/10.1007/BF00420283

[46] Raffaele Quitadamo, Franco Zambonelli, and Giacomo Cabri. 2007. The service ecosystem: Dynamic self-aggregation of pervasive communication services. In *Proceedings of the 1st International Workshop on Software Engineering for Pervasive Computing Applications, Systems, and Environments (SEPCASE'07)*. IEEE, 1–1.

[47] Radin Hamidi Rad, Ebrahim Bagheri, Mehdi Kargar, Divesh Srivastava, and Jaroslaw Szlichta. 2021. Retrieving skill-based teams from collaboration networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2015–2019. DOI: https://doi.org/10.1145/3404835.3463105

[48] Radin Hamidi Rad, Ebrahim Bagheri, Mehdi Kargar, Divesh Srivastava, and Jaroslaw Szlichta. 2022. Subgraph representation learning for team mining. In *Proceedings of the 14th ACM Web Science Conference*. ACM, 148–153. DOI: https://doi.org/10.1145/3501247.3531578

[49] Radin Hamidi Rad, Hossein Fani, Mehdi Kargar, Jaroslaw Szlichta, and Ebrahim Bagheri. 2020. Learning to form skill-based teams of experts. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*. ACM, 2049–2052. DOI: https://doi.org/10.1145/3340531.3412140

[50] Radin Hamidi Rad, Aabid Mitha, Hossein Fani, Mehdi Kargar, Jaroslaw Szlichta, and Ebrahim Bagheri. 2021. PyTFL: A Python-based neural team formation toolkit. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*. ACM, 4716–4720. DOI: https://doi.org/10.1145/3459637.3481992

[51] Radin Hamidi Rad, Shirin Seyedsalehi, Mehdi Kargar, Morteza Zihayat, and Ebrahim Bagheri. 2022. A neural approach to forming coherent teams in collaboration networks. In *Proceedings of the 25th International Conference on Extending Database Technology*. OpenProceedings.org, 2:440–2:444. DOI: https://doi.org/10.48786/edbt.2022.37

[52] Anna Sapienza, Palash Goyal, and Emilio Ferrara. 2019. Deep neural networks for optimal team composition. *Front. Big Data* 2 (2019), 14. DOI: https://doi.org/10.3389/fdata.2019.00014

[53] Mauro Sozio and Aristides Gionis. 2010. The community-search problem and how to plan a successful cocktail party. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 939–948. DOI: https://doi.org/10.1145/1835804.1835923

[54] Qiaoyu Tan, Ninghao Liu, Xing Zhao, Hongxia Yang, Jingren Zhou, and Xia Hu. 2020. Learning to hash with graph neural networks for recommender systems. In *Proceedings of the Web Conference*. ACM/IW3C2, 1988–1998. DOI: https://doi.org/10.1145/3366423.3380266

[55] Hyeongon Wi, Seungjin Oh, Jungtae Mun, and Mooyoung Jung. 2009. A team formation model based on knowledge and collaboration. *Expert Syst. Appl.* 36, 5 (2009), 9121–9134. DOI: https://doi.org/10.1016/j.eswa.2008.12.031

[56] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*. ACM, 495–503. DOI: https://doi.org/10.1145/3018661.3018689

[57] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* 32, 1 (2021), 4–24. DOI: https://doi.org/10.1109/TNNLS.2020.2978386

[58] Yuping Yang, Fiona Mahon, M. Howard Williams, and Tom Pfeifer. 2006. Context-aware dynamic personalised service re-composition in a pervasive service environment. In *Proceedings of the 3rd International Conference on Ubiquitous Intelligence and Computing (Lecture Notes in Computer Science, Vol. 4159)*. Springer, 724–735. DOI: https://doi.org/10.1007/11833529_74

[59] Morteza Zihayat, Aijun An, Lukasz Golab, Mehdi Kargar, and Jaroslaw Szlichta. 2017. Authority-based team discovery in social networks. In *Proceedings of the 20th International Conference on Extending Database Technology*. OpenProceedings.org, 498–501. DOI: https://doi.org/10.5441/002/edbt.2017.54

[60] Armen Zzkarian and Andrew Kusiak. 1999. Forming teams: An analytical approach. *IIE Trans.* 31, 1 (1999), 85–97.

Received 3 April 2022; revised 24 November 2022; accepted 12 March 2023