Fuzzy Systems and Data Mining VIII
A.J. Tallón-Ballesteros (Ed.)

doi:10.3233/FAIA220374

Applied Decision Focused Learning: An End-to-End Decision System for Task Allocation

Chia E. TUNGOM[a,b], Xianfeng DING[a], Lin WANG[a], Changyuan ZHOU[a], Jiawei CHEN[a], Xingxing CHENG[a] and Ji YUAN[a1]
[a] *Onewo Space-Tech Service Co., Ltd., Shenzhen, 518049, China*
[b] *Shenzhen University, Shenzhen 518060, China*

Abstract. Continuous timely repair and replacement of infrastructures, equipment and utilities play an important role in maintaining the smooth-running of a city or local community. Thereby, to help individuals and businesses go about their daily activities with ease, it is vital to develop a proper method for automatically identifying and assigning capable workers for tasks. This paper defines the community management service task allocation problem as CMS-TAP and hence an end-to-end "recommendation + allocation" network, i.e. a task recommender and allocation optimization network (denoted as TROpt-NET), is then developed for handling such problem. TROpt-NET consists of two layers, namely one for predicting worker ability and the other for allocating tasks which are TR Layer and TA Layer, corresponding to "recommendation" and "allocation" of tasks. Different from operations research approaches where workers are assigned to jobs based on their pre-labelled skills and fixed locations, we propose a task recommender and allocation optimization network. The TR layer is a task recommender system designed to learn implicit worker abilities for different tasks using Neural Collaborative Filtering (NCF) by mining a historical dataset of worker task completion. Whereas in the TA layer a differential optimization approach for allocation is used because of its differentiable property and ability to allow for backpropagation to the prediction layer. In this study, we first formulate the CMS-TAP problem as a recommendation +optimization problem and then propose and end-to-end network architecture that tackles the problem in a real-world setting. TROpt-NET curbs uncertainty and assumptions in optimization by learning to more accurately approximate worker ability across different tasks. Additionally, the network can learn implicit worker abilities enabling optimal utilization of workers across a wide range of tasks, which is often ignored in task allocation problems. We find that normalizing worker ability across all tasks improves the implicit learning capability of the network and that good approximations don't always lead to optimal allocation but learning allocations by backpropagating through recommendations improves the allocation objective. Offline experiments on a real-world large-scale dataset demonstrate the effectiveness of our proposed TROpt-NET.

Keywords. Deep learning, optimization, task allocation

[1] Corresponding Author, Ji Yuan, Onewo Space-Tech Service Co., Ltd., Shenzhen, 518049, China; Email: yuanj36@vanke.com

1. Background

There is an increasing need for continuous and timely maintenance and operation of infrastructure/equipment to keep the smooth functioning of residential areas and communities. Allocating workers to suitable tasks in a timely manner increases operational efficiency which goes to improve the quality of service for customers as well as workers' experience[1]. Onewo has strong requirements for task allocation since employees have to provide residential, office, and public areas with efficient and timely maintenance, repair, and replacement services each day.

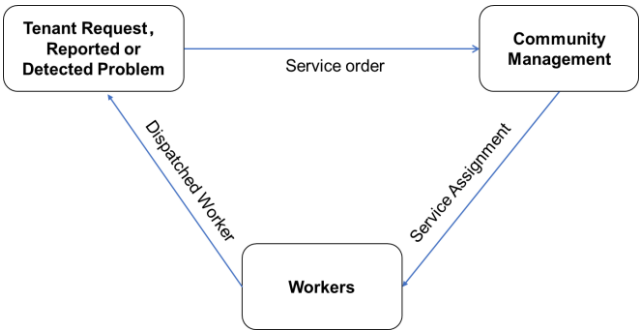

Fig. 1 Task request allocation and execution procedure

Fig. 1 illustrates a typical task request, allocation and execution procedure. A typical process is as follows, a tenant reports a request for a service, and subsequently management finds a suitable worker available and assigns the worker to complete the task. The worker then goes to the service site, completes the task and sends a report via the platform. Suppose the tenant expects the service to be completed as soon as possible, management is responsible for ensuring timely completion of tasks while workers hope suitable tasks are allocated to them. If a worker is assigned a task for which he\she is not highly skilled, it is likely that they will take a longer time to complete the task or might not do a proper job.

This can lead to waste of resources and a short life span for the completed service to be reported again. As a result, learning a worker's ability for tasks is vital for optimal task allocation and operational efficiency which also improves tenant and worker' experience. We denote our task allocations problem as CMS-TAP (Community Management Service Task Allocation Problem).

CMS-TAP is similar to work order scheduling which has widely been studied and solved using operations research optimization models for areas such as scheduling in smart factories [2], facility maintenance [3] and production planning [4]. Automating task allocation is increasingly important owing to complexities and local requirements for rapid maintenance and replacement of electrical, mechanical and electro-mechanical etc. [5].

This study presents an end-to-end system for CMS-TAP that has the potential to automate the identification and allocation of tasks to workers by applying a data mining method to learn worker abilities and therefore assign them to suitable tasks in a timely manner whenever a problem arises or a request is made.

There are three indispensable entities in the CMS-TAP and include: 1) tasks 2) workers 3) management. In our framework, a task is initiated when a tenant or anyone

submits a service request with related details (e.g., problem, location, severity, picture etc.) to management. The assigned worker travels to the task location, completes the tasks and reports back to management when the task is complete. Management keeps digital records of task requests and completion with associated features. One of the key challenges is finding the most suitable workers and assigning them to requested tasks. This process entails selecting appropriate workers from a pool to perform various tasks while satisfying certain constraints.

Management has a database of information with regards to all workers, e.g. work hours, ability, preferences and tasks (location, expertise and domain) and directly assigns workers to appropriate jobs. Allocating workers to tasks is often so as to maximize system cost such as timely completion and quality of service (QoS). Management is tasked to make use of the full set of worker skills and resources in order to ensure tasks are performed efficiently. There are a several difficulties to the problem including the fact that most of the management task is manual, uncertainty in ability of workers across different tasks. Also, an increase in task quantity increases the solution search space along with the constraints of tasks and workers and as a result, the problem becomes more sophistical and non-trivial for a human manager to handle.

Previous works treat the CMS-TAP as an optimization problem where worker capabilities are static and the focus in on allocating workers to tasks. This work is the first attempt to address this problem as far as we know using a machine learning approach. Related studies that optimize task allocation for urban and community management using machine learning approaches is the spatial crowdsourcing (SC) task allocation problem [1]. In the SC scenario, workers are public participants that have signed up to the SC platform while in our setting, workers are employees of a service management company and outside participants cannot take up tasks.

2. Related Work

The CMS-TAP has rarely been studied and existing literature is mainly based on crowdsource task allocation adopting heuristic assignment algorithms. In this section, we will introduce the works related to TROpt-NET, specifically Task Recommendation related to TR layer and Task Allocation related to TA layer.

2.1. Task Recommendation

Existing literature on task recommendation systems gives valuable know-how in the area from various aspects, approaches, architectures and domains, resulting in a vast range of knowledge in many areas[6]. As there are several domains, task recommendation changes with their application and optimization objective and hence we review both from the optimization objective setting and task recommendation. New service models pertaining to advances in mobile and IoT computing in spatial crowdsourcing have seen significant growth where tasks are recommended to workers to choose from using a recommender task allocation system [7]. In crowdsourcing tasks like the Amazon Mechanical Turk [8], Ho et. al. [9] modelled the task assignment problem for which a registered worker has a defined set of tasks with the objective is to allocate workers to tasks so as to maximize for worker efficiency. For micro-task assignment in the spatial crowdsourcing domain, Guo et al. [10] outline a fundamental approach for micro-task assignment, where the main strategy adopted for optimizing maximizes the task quality

of completion and system cost minimization. Cheng et al. [11] proposed a model to dynamically recommend and assign spatial tasks to workers under strict skill, distance, time and budget constraints. In the area of specialized-domain crowdsourcing contests like InnoCentive, Kaggle etc, participants are required to be skilled and interested in the problem. The recommendation approach used will mainly takes into consideration features rich enough to match tasks and participants. To estimate the likeliness of each participant and deliver a ranked recommendation lists of contestants, Baba et al. [12] used previous participation data of solvers and contests to inform their allocation. Given a group of participants in an open contest, Mo et al. [13] proposed a solution that recommends tasks to a worker with high motivation to complete, and the probability of completing a task depends on competitors in the pool for the task. A more comprehensive and structured literature on task recommendation can be found in the study by Xi et al. [6]

2.2. Task Allocation

In the task allocation domain, there are several studies where workers have to travel from their current location to the tasks [14-16]. The growth in the use of smart applications and devices has led to task allocation been studied and applied in both industry and academia. Service delivery companies like Meituan [17], Didi [18] and TaskRabbit [19] all allocate tasks to their workers in a highly dynamic environment. The approach common in industries for task assignment can be classed into algorithmic [20], static matching and dynamic matching and planning [21]. In the modeling approach for matching, allocation is usually modeled such that workers and tasks can be represented using a network bipartite graph. Subsequently, the weight in the edge of a worker connecting to a task represents the cost of that worker completing the task and the goal is to find the best match in a graph between workers and tasks [22,23]. In the planning methodology, a task assignment takes into consideration the route and plans a route to be taken by a worker to complete a sequence of tasks [21].

For complex task assignments, existing approaches assign a task to a small set of workers who meet given constraints for skills. Another approach to the problem is to break a bigger problem into a group of many smaller simple subtasks then select and allocate a qualified worker to a subtask [24]. However, those approaches are beyond our study in this paper.

3. Task Modelling

In this section we present the CMS-TAP as a recommendation-allocation problem.

3.1. Problem Formulation

In the recommendation-allocation scenario, an optimization problem can be modelled as shown in Eq. (1) where the objective function depends on θ_{true} a parameter assumed to be known. Notice that, θ_{true} can be the time taken for a worker to complete a task and is often unknown.

$$y = \min_{x*feasible} f(X, \theta_{true}) \dots (1)$$

Given a dataset D with features ξ we can learn and approximate θ_{true} by mining historical worker behavior. The ultimate objective is to select an optimal decision (the decision variable X), specifically in our case, an optimal task allocation $X^*(\xi)$ which is a function of the features. By learning X from the dataset features ξ and estimating θ_{true} from the dataset D, our objective can be written as:

$$\min_{x^* feasible} E_{(\xi,\theta_{true})} \sim D[f(X^*(\xi), \theta_{true})] \ldots (2)$$

The goal is to learn a model $\Phi(.,w)$ which accurately estimates a missing parameter θ_{true} using the features ξ. We can then infer θ_{true}, which we then use to solve the given optimization problem to find optimal variables for X. $x^* feasible$ is the feasible space of our decision variables where all constraints are met. For instance, a worker cannot be allocated two tasks simultaneously to be to be taken up at the same time or two workers cannot be assigned to a single task that requires only a single worker.

Our decision problem can be broken down into two parts:

1. Learn a model $\Phi(.,w)$ using given features ξ and accurately estimate θ_{true}.
2. Use estimated parameters θ_{true} to find optimal values of X that minimize the optimization objective $f(X, \theta_{true})$.

In our case 1) is related to the TR layer and 2) is related to the TA layer, these layers are differentiated as shown in Fig. 2.

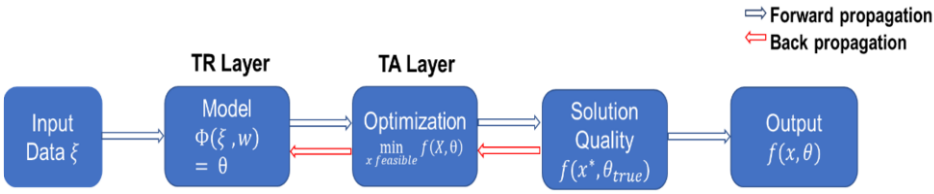

Fig. 2 Recommendation + Optimize Framework

3.2. CMS-TAP

We consider a community/urban management service platform in which there are K tasks and L workers during a certain time period. Management is tasked to assign tasks to workers at each time $n \in N$. Each worker has different abilities (time to complete a task) for different tasks. The worker's ability for task $k \in K$ is the probability $p_{l,k}$ of completing a task k in the shortest time possible. We let $A_{l,k} = 1$ if worker $l \in L$ is assigned the k^{th} task from their recommended set K, and $A_{l,k} = 0$ otherwise, we let $X_{l,k} = \theta_{l,k}$ be the time taken by workers l to complete task k.

We model the task allocation problem as a recommendation-allocation problem. Since $\theta_{l,k}$ is unknown and needs to be estimated. We use a recommender approach, particularly collaborative filtering because implicit workers abilities can be learned. We formulate our optimization problem as stated in Eq. (3).

$$f(X, \theta) = \sum_{k=1}^{K} . \sum_{l=1}^{L} A_{l,k}\theta_{l,k} \ldots (3)$$

where $A_{l,k} \in \{0,1\}$ denotes whether worker l is assigned a task k or not. Our constraints for allocation include (1) the worker must be on duty (2) the assigned task must be within the worker's community or jurisdiction of work.

In our implementations $p_{l,k}$ is used in place of $\theta_{l,k}$ because it brings several advantages including interpretability and uniformity to task completion times, e.g. it can take 5

minutes to repair a light bulb and one hour to repair a leaking pipe. $p_{l,k}$ is only concerned with the probability of a worker completing the task in a predetermined shortest possible time which makes the data uniform across all tasks rather than having highly skewed data. Eq. (3) is the objective function for the TA layer in our network. The TR layer solves a regression problem by predicting the time taken for a worker to complete a task. In the TR Layer our output is a probability which correspond to the likelihood of a worker completing a task in the shortest possible time ever completed for each particular task.

3.3. TROpt-NET

We first present the overall framework for TROpt-NET, elaborating on how we use NCF in the TR layer to implicitly learn worker ability which is then utilized by the TA layer to learn an optimal allocation.

- TR Layer

The main objective of the TR layer is to learn an unknown parameter $\theta_{l,k}$, a worker's ability to complete any specific task. Pertaining to our model requirements, we adopt a NCF approach to learn user ability because of its ability to learn implicit preferences and also because it has been extensively studied for and applied to the reality recommender systems [25-27].

In the TA layer for our NCF model, the input features or data include workers, tasks, worker-related features like age, years of experience, city of residence, rating (after task completion) etc., and completion times for tasks. We ignore context features in this study. The task completion time which we refer to as worker ability is the target feature for the NCF algorithms which we aim to estimate. We normalize completing task k by worker l to be in the range [0,1], the probability $p_{l,k}$ of completing can be computed as Eq. (4).

$$p_{l,k} = 1 - \frac{l_k - l_{min}}{l_{max} - l_{min}} \dots (4)$$

where l_{max} and l_{min} the maximum and minimum completion time, respectively. The normalized l_k which is the probability $p_{l,k}$ is used as our target feature for the NCF model. We can say the probability of a worker l to complete a task k in l_{min} time is $p_{l,k}$. We use l_{min} because a higher probability of the shortest completion time is desired.

The NCF consists of a Generalized Matrix Factorization (GMF) that models latent feature interactions and an MLP that learns the interaction function from the data. The GMF and MLP are fused in the last layer and a sigmoid function is applied to obtain the predicted score or worker ability as seen in the architecture in **Fig. 3**. GMF which is an embedding vector of worker (task) interaction is obtained by applying a linear kernel to a one-hot encoded worker (task) input feature. Consider the worker latent vector to be p_w and task latent vector to be q_t, the mapping function that produces the GMF is given by the element-wise product of vectors shown in Eq. (5).

$$\phi_1(p_w, q_t) = p_w \odot q_t \dots (5)$$

The MLP consists of two sub MLP networks. One network uses worker and tasks vectors while the other uses worker features. The two MLP layers features are fused before producing a unified final layer for MLP which from our experiments works best. The worker-task MLP layer can be modelled with Eq. (6):

$$\phi_2(p_w, q_t) = a_i(W_i^T(p_w, q_t) + b_i)) \dots (6)$$

where a, W and $b$are the activation function, weight matrix and bias vector. The worker feature MLP sub-layer uses worker context features denoted as c_w. Worker features

which add context information about workers essentially enriching the network is then used in the sub-layer to learn worker behavior. The sublayer is modelled as shown in Eq. (7):

$$\phi_3(p_w, q_t, c_w) = a_i(W_i^T(p_w, q_t, c_w) + b_i)) \ldots (7)$$

The complete network architecture for the NCF is as shown in **Fig. 3.** In the final phase, the MLP and GMF are fused to obtain output features and then apply a sigmoid function to get predicted scores or abilities for the workers.

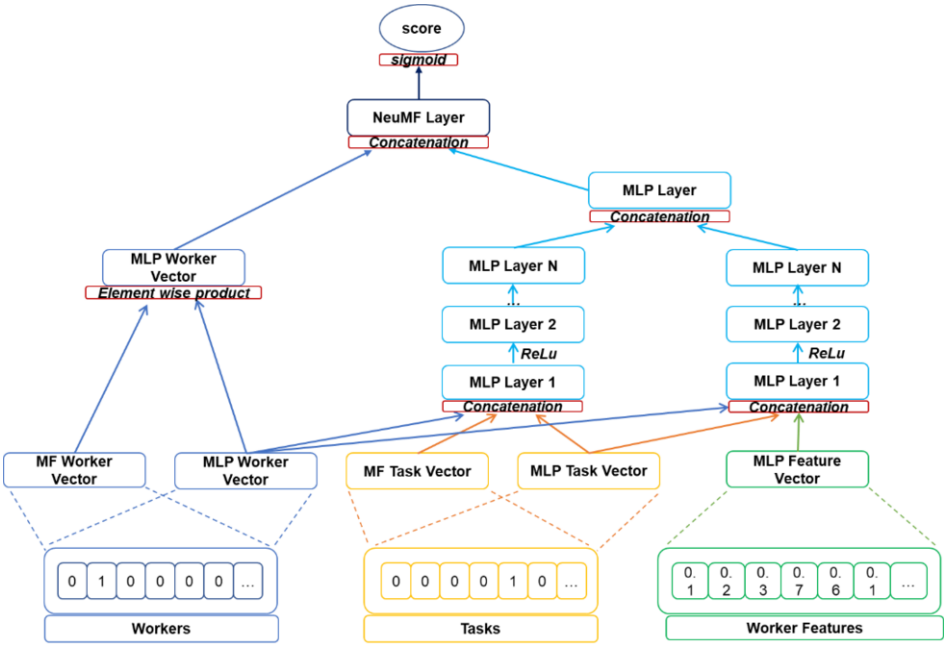

Fig. 3 NCF Architecture for TR Layer

- TA Layer

The main objective of this layer is to learn an optimal allocation to minimize cost (overall time to complete tasks). Unlike traditional optimization approaches that directly minimize and an objective function, this layer depends on the learned worker abilities $\theta_{l,k}$ and requires a gradient-based end-to-end learning approach as seen in decision-focused learning [28-29] methods. In principle, this end-to-end approach achieves solutions of superior quality to its two-stage learning subordinate where the predicted unknown is directly used for optimization. In the decision focused learning approach, we learn $\theta_{l,k}$ which helps with not just our prediction quality but also with our optimization quality.

We use differential optimization to iteratively search for optimal allocation of workers to tasks by back-propagating to the TR layer where learning the unknown worker ability θ_{true} is updated. We calculate the derivative of our outcome and evaluate on the known optima **X** and parameter θ_{true}. We then apply the chain rule on the weights of the model with respect to the derivative of the solution i.e.:

$$\frac{df(x^*, \theta_{true})}{dw} = \frac{df(x^*, \theta_{true})}{dw} \frac{d(x^*)}{d\theta} \frac{d\theta}{dw} \ldots (8)$$

where $\frac{df(x^*,\theta_{true})}{dw}$ differentiates for the optimal allocation, $\frac{d(x^*)}{d\theta}$ is obtained by differentiating through the KKT condition of our optimization problem and $\frac{d\theta}{dw}$ optimizes for the model weights of the NCF in the TR layer.

The end-to-end approach using differential optimization automatically learns worker ability and allocation quality by back-propagating through the TA Layer and TR layer and updating model weights w, worker abilities θ and allocation solution X as formulated in Eq. (7). This approach directly minimizes our objective function in Eq. (3) by learning an allocation from previous assignments and worker ability from historic tasks completed.

4. Experiments

We evaluate the competitiveness of our proposed methodology using mean squared error (MSE) and regret. MSE is commonly used in regression problems to evaluate the quality of predictions by taking the mean difference between predicted and real values for each sample in the dataset. Regret is the difference in the solution found after estimating the unknown parameter θ by observation. Note that our evaluation of regret is solely on the optimization quality. It can be inferred that prediction quality plays a role in the process of evaluating our regret.

We compare the performance of our algorithm with a decision focussed learning method, two-stage learning and surrogate assisted decision focussed learning method. We in essence present a fair comparison of our method for the task allocation problem with other state-of-the-art methods.

Decision-Focused Learning-NCF: We denote this model as **DFL-NCF** which is a decision focused learning method taken from the study [27] with NCF in the TR layer and takes only workers and tasks as input features. The input features are based on the NCF architecture.

Two-Stage Learning-NCF: We denote the model as **TSL-NCF** which is the two-stage learning approach adapted from the study [29] with NCF in the TR layer for predicting worker ability. In this network, the NCF model takes only workers and tasks as input features.

Surrogate Assisted Learning-NCF: We denote this model as **SAL-NCF.** It is a decision focused learning method with a surrogate model in the TA Layer and parameter reparameterization before the TA Layer as outlined in the study [29]. The input features here are workers and tasks based on NCF design.

Two-Stage Learning-TROpt-Net (TSL-NCF-T): Uses our proposed NCF architecture with a two-stage optimization approach taking worker, tasks and worker features as input. We use the short form **TSL-NCF-T** to represent the approach**.**

Surrogate Assisted Learning-TROpt-Net (SAL-NCF-T): Uses our proposed NCF architecture with a surrogate TA layer with reparameterization before and taking worker, tasks and worker features as input. We denote the methodology as **SAL-NCF-T**.

4.1. Data Processing and Model Setting

Our dataset is urban and community task completion data in Shenzhen queried between January to December of the year 2020. The dataset consists of a total of 299 workers and

230 unique tasks classified into 12 categories. For processing the data, we normalize the completion time of tasks (target variable) using Eq. (4). Worker related features are normalized using Eq. (4) while categorical variables are one hot encoded. The dataset is then split into 60% training set, 10% validation and 30% testing.

In our model, Neural Collaborative Filtering [16] is used to learn worker ability. We design the architecture so as to learn embeddings for every worker and task. The abilities are calculated by adding a concatenation of worker, task and worker-task embedding to a network having fully connected layers as shown in **Fig. 3**. To train GMF and MLP, we choose to use the Adam optimizer because it is capable of adapting the learning rate for parameters and enables faster convergence while automatically tuning the learning rate. After the NeuMF (Neural Matrix Factorization) a vanilla SGD with a learning rate of 0.001 is used because this layer is not suitable for methods based on momentum like Adam. A total of 4 MLP layers in the MLP sub modules are used with the ReLU activation function before concatenation as seen on the right in **Fig. 3**. The settings for architecture related to this study are presented in Table 1:

Table 1. Performance of algorithms

Metric	DFL-NCF	TSL-NCF	SAL-NCF	TSL-NCF-T	SAL-NCF-T	TROpt-Net
Train Loss	0.294	0.277	0.311	0.217	0.301	0.272
Validation Loss	0.300	0.283	0.342	0.222	0.322	0.277
Test Loss	0.365	0.299	0.351	0.218	0.333	0.288
Train Regret	2.201	3.369	2.112	1.991	1.422	0.918
Validation Regret	2.697	3.791	2.784	1.998	1.938	1.643
Test Regret	2.641	3.773	3.111	2.531	2.226	1.820

4.2. Results Analysis

Our methodology clearly outperforms all other methods in the optimization regret which is the allocation objective but doesn't give the best results for the loss function which optimizes for prediction quality as seen in Tab. 1. This is common behavior for decision focused learning methods and is normally because the prediction quality doesn't always inform good decisions but aids in good decisions. We also observe that methods that use our proposed TA layer architecture NFC with worker features perform better. The TA layer we proposed is more informed about workers and so we see it making better predictions as seen with TSL-NCF-T than all other methods. Although prediction quality doesn't always lead to better decisions, we see that it clearly helps as the performance of TSL-NCF-T and SAL-NCF-T on the regret show significantly better results from the non-worker feature informed models.

Overall the proposed NCF approach in the TA layer of TROpt-Net leads to better decisions in the optimization objective and better predictions for the two-stage learning method. Therefore, we not only improve the decision quality but also the prediction quality of our model.

Our findings show that machine learning methods can be competitive with specialized mathematical optimization tools for NP-complete problems like the task allocation problem. Unlike mathematical optimization methods that assume unknown parameters like worker ability, an end-to-end method like our proposed network makes no assumptions and can automatically learn and approximate worker ability across all

tasks reducing uncertainty in allocation and enabling a more robust system for task allocation as demonstrated in this study. To reduce uncertainty, most optimization methods often decompose a problem by breaking it down into smaller ones enabling faster convergence to a more accurate solution [5]. Decomposing the problem requires expert knowledge and even so optimal results cannot be guaranteed. In this study, we make no assumptions and decomposition of our problem which is advantageous for end users who sometimes do not have expert knowledge of formulating an optimization problem.

5. Conclusion

In this study, we focus on a task allocation problem in a community/urban management setting. We model our problem as a recommendation and allocation problem and then design a network using a Neural collaborative filtering and differential optimization for the prediction of worker ability and task allocation respectively. The main challenge we face in our solution design method is on how to model task allocation as a recommendation problem that learns implicit worker ability. We treat a worker completion time for a task using a normalized probability of completing a task within the shortest time for a specific task from the dataset. Empirically, with a fair comparison on similar approaches, our proposed method outperforms other decision focused and two-stage learning methods owing to the adaptation and proposed network architecture. Unlike like in conventional optimization problems where task allocations optimization is carried out without the need for worker data and performance labels, the proposed method can only be utilized in settings where rich worker data is available with ratings available for completed tasks. The major shortcoming and applicability of this study is the data requirements. It is well known that a machine learning model is only as good as its data. If the dataset is too noisy, it might turn too hard for the network to approximate worker ability and make good recommendation which ultimately leads to suboptimal or poor results. Having as accurately labelled dataset of worker ability is key to fully utilizing the potential of this optimization approach. An area we did not explore in this research was the use or performance of a mathematical optimizer in our TA layer. In the future we plan to utilize widely studied and developed operations research optimizer for our TA which has the potential for good allocations given the problem space can be reduced by our recommender algorithm. This study shows machine learning and optimization methods can produce superior results for task allocation problem.

References

[1] Jiao, Yuxin, et al. "A Fine-Grain Batching-Based Task Allocation Algorithm for Spatial Crowdsourcing." ISPRS International Journal of Geo-Information 11.3 (2022): 203.
[2] Herrmann, Jeffrey W. "Information flow and decision-making in production scheduling." IIE Annual Conference. Proceedings. Institute of Industrial and Systems Engineers (IISE), 2004.
[3] Chen, Weiwei, et al. "BIM-based framework for automatic scheduling of facility maintenance work orders." Automation in Construction 91 (2018): 15-30.
[4] Zhou, Tong, et al. "Multi-agent reinforcement learning for online scheduling in smart factories." Robotics and Computer-Integrated Manufacturing 72 (2021): 102202.
[5] Purwar, Anupam. "Automated Planning Tool (APT): A mixed integer non-linear programming problem solver for Workorder scheduling."

https://assets.amazon.science/35/14/3b6c50574f8ca110e1c702e8e44a/automated-planning-tool-apt-a-mised-interger-non-linear-programming-problem-solver-for-workorder-scheduling.pdf
[6] Yin, Xicheng, et al. "Task recommendation in crowdsourcing systems: A bibliometric analysis." Technology in Society 63 (2020): 101337.
[7] Musthag, Mohamed, and Deepak Ganesan. "Labor dynamics in a mobile micro-task market." Proceedings of the SIGCHI conference on human factors in computing systems. 2013
[8] Chen, Jenny J., et al. "Opportunities for crowdsourcing research on amazon mechanical turk." Interfaces 5.3 (2011): 1.
[9] Ho, Chien-Ju, and Jennifer Vaughan. "Online task assignment in crowdsourcing markets." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 26. No. 1. 2012.
[10] Guo, Bin, et al. "Task allocation in spatial crowdsourcing: Current state and future directions." IEEE Internet of Things Journal 5.3 (2018): 1749-1764.
[11] Chen, Huihui, et al. "A generic framework for constraint-driven data selection in mobile crowd photographing." IEEE Internet of Things Journal 4.1 (2017): 284-296.
[12] Baba, Yukino, Kei Kinoshita, and Hisashi Kashima. "Participation recommendation system for crowdsourcing contests." Expert Systems with Applications 58 (2016): 174-183.
[13] Mo, Jiahui, Sumit Sarkar, and Syam Menon. "Know when to run: Recommendations in crowdsourcing contests." MIS quarterly 42.3 (2018): 919-944.
[14] Liang, Decui, et al. "A novel approach of two-stage three-way co-opetition decision for crowdsourcing task allocation scheme." Information Sciences 559 (2021): 191-211.
[15] Zhang, Xiaoyu, et al. "Privacy-preserving and verifiable online crowdsourcing with worker updates." Information Sciences 548 (2021): 212-232.
[16] Xu, Wenqiang, Liangxiao Jiang, and Chaoqun Li. "Improving data and model quality in crowdsourcing using cross-entropy-based noise correction." Information Sciences 546 (2021): 803-814..
[17] Meituan (2022). https://www.meituan.com/. Accessed May 4, 2022
[18] Didichuxing (2022). https://www.didiglobal.com/. Accessed May 4, 2022
[19] Taskrabbit (2022). http://www.taskrabbit.com/. Accessed May 4, 2022
[20] Yongxin Tong, Zimu Zhou, Yuxiang Zeng, Lei Chen, Cyrus Shahabi, Spatial crowdsourcing: a survey, The VLDB Journal 29 (1) (2020) 217–250.
[21] Asghari, Mohammad, and Cyrus Shahabi. "On on-line task assignment in spatial crowdsourcing." 2017 IEEE International Conference on Big Data (Big Data). IEEE, 2017.
[22] Tong, Yongxin, et al. "Spatial crowdsourcing: a survey." The VLDB Journal 29.1 (2020): 217-250.
[23] He, Shibo, et al. "Toward optimal allocation of location dependent tasks in crowdsensing." IEEE INFOCOM 2014-IEEE Conference on Computer Communications. IEEE, 2014.
[24] Ni, Wangze, et al. "Task allocation in dependency-aware spatial crowdsourcing." 2020 IEEE 36th International Conference on Data Engineering (ICDE). IEEE, 2020.
[25] Perifanis, Vasileios, and Pavlos S. Efraimidis. "Federated Neural Collaborative Filtering." Knowledge-Based Systems 242 (2022): 108441.
[26] Hiriyannaiah, Srinidhi, Siddesh GM, and K. G. Srinivasa. "DeepLSGR: Neural collaborative filtering for recommendation systems in smart community." Multimedia Tools and Applications (2022): 1-20.
[27] Long, Lianjie, et al. "Multi-task learning for collaborative filtering." International Journal of Machine Learning and Cybernetics 13.5 (2022): 1355-1368.
[28] Mandi, Jayanta, Peter J. Stuckey, and Tias Guns. "Smart predict-and-optimize for hard combinatorial optimization problems." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34. No. 02. 2020.
[29] Wang, Kai, et al. "Automatically learning compact quality-aware surrogates for optimization problems." Advances in Neural Information Processing Systems 33 (2020): 9586-9596.