Who will Win the Data Science Competition? Insights from KDD Cup 2019 and Beyond

HAO LIU and QINGYU GUO, Hong Kong University of Science and Technology
HENGSHU ZHU, Baidu Talent Intelligence Center, Baidu Inc.
FUZHEN ZHUANG, Institute of Artificial Intelligence, Beihang University and SKLSDE, School of Computer Science, Beihang University
SHENWEN YANG and DEJING DOU, Baidu Inc.
HUI XIONG, Hong Kong University of Science and Technology

Data science competitions are becoming increasingly popular for enterprises collecting advanced innovative solutions and allowing contestants to sharpen their data science skills. Most existing studies about data science competitions have a focus on improving task-specific data science techniques, such as algorithm design and parameter tuning. However, little effort has been made to understand the data science competition itself. To this end, in this article, we shed light on the team's competition performance, and investigate the team's evolving performance in the crowd-sourcing competitive innovation context. Specifically, we first acquire and construct multi-sourced datasets of various data science competitions, including the KDD Cup 2019 machine learning competition and beyond. Then, we conduct an empirical analysis to identify and quantify a rich set of features that are significantly correlated with teams' future performances. By leveraging team's rank as a proxy, we observe "the stronger, the stronger" rule; that is, top-ranked teams tend to keep their advantages and dominate weaker teams for the rest of the competition. Our results also confirm that teams with diversified backgrounds tend to achieve better performances. After that, we formulate the team's future rank prediction problem and propose the *Multi-Task Representation Learning* (MTRL) framework to model both static features and dynamic features. Extensive experimental results on four real-world data science competitions demonstrate the team's future performance can be well predicted by using MTRL. Finally, we envision our study will not only help competition organizers to understand the competition in a better way, but also provide strategic implications to contestants, such as guiding the team formation and designing the submission strategy.

CCS Concepts: • **Information systems → Data mining;**

Additional Key Words and Phrases: Data science competition prediction, deep representation learning, multi-task learning

This work is supported by the National Natural Science Foundation of China under Grant No. 62102110 and 62176014.
Authors' addresses: H. Liu (corresponding author), Q. Guo, and H. Xiong (corresponding author), Thrust of Artificial Intelligence, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, 511400, Guangdong, China and Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China; emails: liuh@ust.hk, qguoag@connect.ust.hk, xionghui@ust.hk; H. Zhu (corresponding author), Baidu Talent Intelligence Center, Baidu Inc., Beijing, 100085, China; email: zhuhengshu@baidu.com; F. Zhuang, Institute of Artificial Intelligence, Beihang University and SKLSDE, School of Computer Science, Beihang University, Beijing, 100085; email: zhuangfuzhen@buaa.edu.cn; S. Yang and D. Dou, Baidu Inc., Beijing, 100085, China; emails: {yangshengwen, doudejing}@baidu.com.

1556-4681/2022/04-ART98 $15.00
https://doi.org/10.1145/3511896

ACM Reference format:
Hao Liu, Qingyu Guo, Hengshu Zhu, Fuzhen Zhuang, Shenwen Yang, Dejing Dou, and Hui Xiong. 2022. Who will Win the Data Science Competition? Insights from KDD Cup 2019 and Beyond. *ACM Trans. Knowl. Discov. Data.* 16, 5, Article 98 (April 2022), 24 pages.
https://doi.org/10.1145/3511896

1 INTRODUCTION

In recent years, data science competitions have become a successful way to deliver state-of-the-art solutions to commercial and non-profit institutions (the organizer) as well as for data science experts and beginners (the contestant) to sharpen their data science skills. As a result, online data science competition platforms such as Kaggle [28], Tianchi [55], and Dianshi [17] are becoming increasingly popular. Figure 1 gives an illustrative example of the multi-modal transportation recommendation [35] competition we hosted on the Baidu–Dianshi platform. In general, the common routine of the data science competition is a tripartite game, where: (1) the organizer posts a problem with a set of pre-specified competition rules, datasets, as well as awards (such as money, plague, and even a job position); (2) the contestants form a team to compete for the award by submitting state-of-the-art solutions of the problem; and (3) the platform provides a series of functions, including team collaboration, submission evaluation, open discussion, and perhaps computational resources (for example, virtual machine and GPU).

Although there is increasingly more literature sharing their winning solutions of different data science competitions, only a little attention has been paid to the data science competition itself. For example, Athanasopoulos et al. [3] identified the positive effect of the live leaderboard on improving the accuracy of the result, Tausczik et al. [53] found open sharing in data science competition is useful to improve individual but not collective performance, and Lu et al. [39] analyzed the social network attributes among contestants across multiple data science competitions. Nevertheless, existing studies mainly focus on the effects of different competition mechanisms (such as reputation system [16], open discussion [53]) to the final solution quality. Little effort has been made to inspect the team's performance during the competition.

In this article, we investigate the team's time-evolving competition performance in a crowd innovation context [6]. Specifically, we aim at addressing the following two **research questions** (**RQs**) to profile and predict the performance of each team.

- **RQ 1**. What are the representative features that can help to profile team performance in the data science competition?
- **RQ 2**. Can we predict the team's future performance by exploiting significant features found in RQ 1?

Understanding the above two RQs can be beneficial for each party in the game. On the one hand, it is helpful for the organizer and the platform to optimize the competition to engage contestants for crowd innovation and proactively manage the competition, such as resource allocation and fraud detection. On the other hand, it has strategic implications to contestants such as the team formation guidance, the submission strategy design, and the competition participation selection. However, two major challenges arise to answer the above two RQs. First, there is a vast amount of potential features that may be correlated with the team's performance. The first challenge is how to identify the representative features, and to what extent are they correlated with the team's future performance. Second, the future performance of each team is not only correlated with static features (for instance, the number of team members) but also correlated with dynamic features (for example, the number of submissions in each day) that are changing

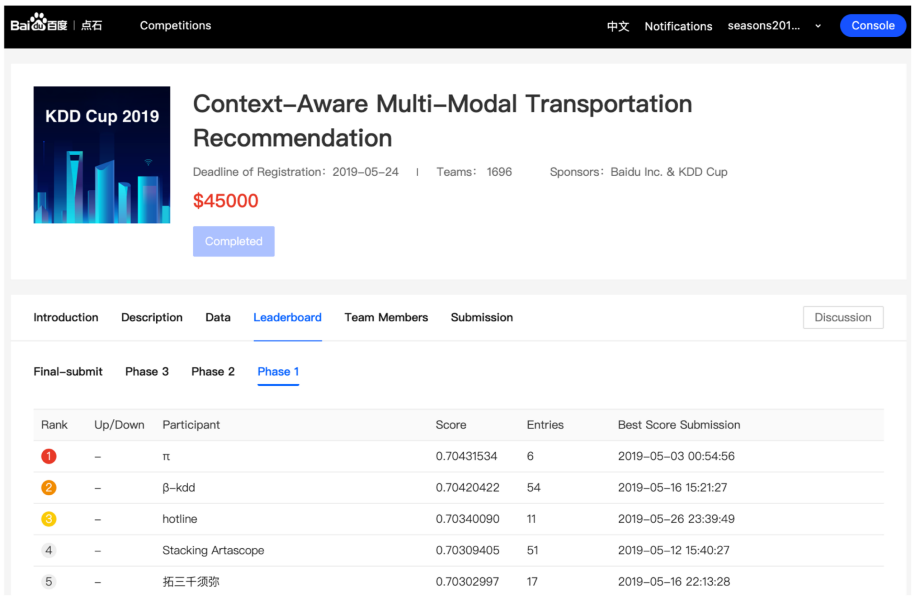


Fig. 1. Interface of the KDD Cup 2019 regular machine learning competition on the Baidu–Dianshi platform. As a full-fledged online platform, Dianshi provides a broad range of functions, such as data acquisition, cloud workbench, result submission, automatic evaluation, live leaderboard, and open discussion. In past years, Dianshi has hosted a series of famous data science competitions such as KDD Cup 2019 regular machine learning competition, WSDM Cup 2019, and so on.

during the competition. Besides, the team's future rank prediction problem suffers from a serious data scarcity problem because data science competitions are usually launched in a short time period (usually less than two months), and the contestant size is relatively small (mostly less than a thousand). How to capture and generalize the non-linear dependency among these correlated features for the team's future performance prediction is another challenge.

To tackle the above challenges, in this article, we leverage team's rank as a proxy, and conduct a data-driven analysis to dissect features that are correlated with the team's future performance. Specifically, we acquire heterogeneous datasets from both the competition platform and third-party professional websites such as Github. From heterogeneous datasets, we extract a rich set of features and quantitatively analyze their relevance with the team's rank. As a result, statistically significant features are used for team performance profiling and subsequent prediction tasks. Furthermore, we propose a ***Multi-task Representation Learning*** (**MTRL**) framework for team's future rank prediction. MTRL integrates both static and dynamic features into a hierarchical multi-task learning framework and derives differentiable team representations by incorporating multiple auxiliary tasks. By sharing common information in multiple related tasks, the predictive model obtains a notable performance improvement on various datasets.

The contributions of this article are summarized as follows. (1) We construct multiple data science competition datasets and quantify a rich set of features that are correlated with team future performance. Our assessments point out that static features such as nation, team size, and dynamic features such as submission count, historical rank are significantly correlated with the team's performance. We also observe "the stronger, the stronger" rule in the competition and confirm the positive effect of team diversity for team's future rank. (2) We formulate the team's future rank prediction problem and propose the MTRL framework. To the best of our knowledge, MTRL is

Table 1. Statistics of Datasets

Data description	KDDCup	Billboard	RemoteSense	WSDMCup
# of submissions	11,957	2,882	2,873	385
# of teams	1,696	1,139	537	136
# of contestants	2,403	1,666	659	155
# of stages	3	3	2	1
Duration (day)	44	38	46	23
Metric	F1 score	MAP	Accuracy score	AUC score

the first deep learning-based framework for predicting the team's future rank in the data science competition. Note the proposed data-driven framework and the prediction model are general and can be extended to incorporate more data sources. (3) We evaluate MTRL on four real-world data science competitions. The results demonstrate the predictability of the team's future performance based on our constructed features, and validate the effectiveness of MTRL against five baselines.

The rest of this article is organized as follows. We describe four real-world datasets used in our study in Section 2, extract and analyze features in Section 3, and elaborate on the MTRL model in Section 4. Experiments of MTRL on four real-world datasets are presented in Section 5. We discuss the key findings, implications, and limitations of this work in Section 6, and review related work in Section 7. Finally, we conclude in Section 8.

2 DATA DESCRIPTION

This section introduces datasets used in this work, including the *competition information data*, the *submission record data*, and the *contestant information data*. We selected four data science competitions at different scales, i.e., KDD Cup 2019 (KDDCup), Billboard Recognition 2018 (Billboard), Remote Sensing 2019 (RemoteSense), and WSDM Cup 2019 (WSDMCup)). All competitions were hosted on Baidu–Dianshi platform.[1] Note that some competitions contain several stages (such as preliminary, semi-final, and final), we excluded late stages where only a small portion of teams participate. The statistics of each dataset are summarized in Table 1. Note that Table 1 presents the number of original registered teams and contestants, where some of them may never submit to the corresponding competition. More detailed analysis are presented in Sections 2.2 and 2.3.

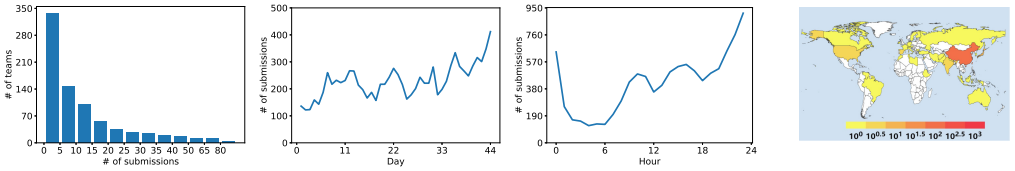
2.1 Competition Information Data

Competition information data describe the basic information and rules of each data science competition. For basic information, each record contains a competition ID, the award information, a list of competition stage IDs, as well as the start and end date of each competition stage. For competition rules, each record contains the maximum number of team members, the maximum number of submission times, the team merge rule, and the evaluation metric. In most data science competitions, each contestant requires to join one team, and each team contains at least one contestant. All our three competition datasets follow this setting.

2.2 Submission Record Data

Submission record data contain the team submission behaviors and historical performances. Once a team submits a new model or result, the Dianshi platform automatically evaluates the submission, and returns a result, such as F1 score, accuracy score, and so on. As a result, each submission generates a submission record data. For instance, [84, 1d30a3, 2019-04-17 08:03:34,1,

[1]All competition information is available at https://dianshi.baidu.com/competition.



(a) Distribution of team submission. (b) # of submissions by day. (c) # of submissions by hour. (d) Nation distribution.

Fig. 2. Data distributions of the KDDCup dataset: (a) the distribution of team submission count; (b) the day distribution of submissions; (c) the hour distribution of submissions; and (d) the distribution of contestant nationality.

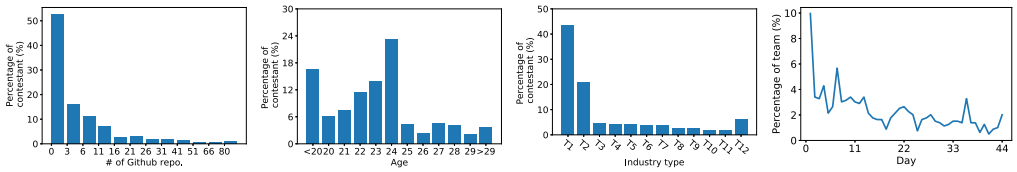
0.03711342] is a submission record, where "84" is the competition id, "1d30a3" is the contestant id, "$2019-04-17 \quad 08:03:34$" denotes the submission time, and "0.03711342" stands for the submission score. The KDDCup dataset contains 11,957 submission records, the Billboard contains 2,882 submission records, the RemoteSense dataset contains 2,873 submission records, and the WSDMCup dataset contains 385 submission records. Each submission record consists of a competition stage ID, a team ID, the submission time, the evaluation time, and the evaluation result.

We briefly explain the distributions of submission records. As illustrated in Figure 2(a), the distribution of team submission numbers nearly follows the power law [14], where over 80.1% teams submit less than 10 times. In fact, a large portion of contestants registered but never submitted a result, and we excluded them from following statistical analysis and prediction task. Figure 2(b) depicts the temporal distribution of submissions and Figure 2(c) depicts the hour distribution of submissions. As can be seen, there are more submissions near competition end, and midnight is the most popular submission time.

2.3 Contestant Information Data

Contestant information data summarize the basic characteristics of each contestant, such as contestant social attributes and professional expertise. In this article, contestant information data w1ere collected from two sources, (1) contestant-provided attributes when registered on Dianshi platform, and (2) Github profiles we linked via the e-mail address. For the contestant's social attributes, each contestant information record contains the gender, the age, the nation, the educational level, the industry type, and the competition registration date. For the contestant's Github attributes, each record contains the number of repositories, the number of followers, and the number of following. Each contestant information record is also associated with a team ID to link with their corresponding teams. For instance, [f7110d, yingshierdu, HKUST, 2019-04-23 21:19:47, employee, PDD, Math, master] is a contestant information record, where "$f7110d$" is the contestant id, "*yingshierdu*" represents the team name, "*HKUST*" denotes the graduation school, "$2019-04-2321:19:47$" corresponds to the participation time, "*employee*" stands for the occupation, "*PDD*" is the associated company, "*Math*" represents the major, and "*master*" denotes the education degree. The KDDCup involves 2,403 contestants, the Billboard includes 1,666 contestants, the RemoteSense has 659 contestants, and the WSDMCup has 155 contestants. Note that for privacy concerns, all records were anonymized and cannot be associated with sensitive information such as names and phone numbers.

We further explain the distributions of contestant information data in KDDCup. The nation distribution is shown in Figure 2(d). As can be seen, China, the United States, and Japan are the top countries contestants come from. Besides, only a few contestants come from Africa, eastern Europe, and mid-east. In fact, the nation distribution of contestants in KDDCup nearly matches



(a) # of Github repositories. (b) Age distribution. (c) Industry type distribution. (d) Join date distribution.

Fig. 3. Data distributions of the KDDCup dataset (continued): (a) the distribution of Github repositories; (b) the distribution of contestant agesand the distribution of jobs; (c) the distribution of industry type; and (d) the distribution of join date.

Table 2. Industry Type List

Type ID	Industry name	Type ID	Industry name
T1	IT	T7	Transportation
T2	Education	T8	Machinery manufacturing
T3	Financial industry	T9	Energy industry
T4	Medical industry	T10	Tourism industry
T5	Public administration	T11	Construction industry
T6	Daily chemical industry	T12	Others (Legal industry, Textile industry, Catering industry, Entertainment, Marketing, Agriculture, and Automotive)

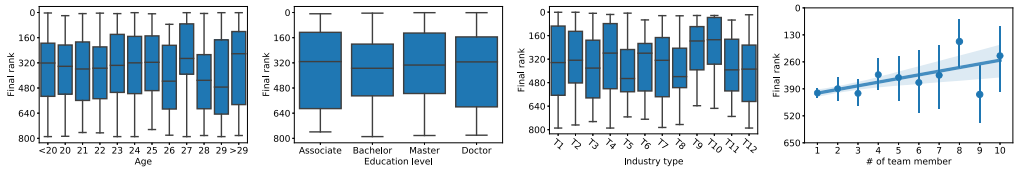
the nation distribution of published AI patents in 2019 [48]. Figure 3(a) reports the distribution of contestant Github repository number. Similar to the distribution of submission count, we observe over 77.8% contestants have less than 10 Github repositories. Besides, we observe 30.1% contestants have a Github account but no repository, which indicates a notable portion of contestants may not be a professional IT engineer. Figure 3(b) plots the distribution of contestant age. We observe most KDDCup contestants are between 20 and 29 years old, which matches our expectation that most contestants are students and junior engineers. Figure 3(c) further illustrates the distribution of contestant industry type, with the detailed industry list is given in Table 2. We merge rare classes and preserve 12 industry types in total. As can be seen, over 35.7% contestants do not work in computer science related industries (such as financial, medical, etc.), indicating that data science competition successfully attracted participants from a broad range of backgrounds. Lastly, Figure 3(d) depicts the contestant register date distribution. Overall, the initial period attracts most contestants and much fewer contestants join near the competition end.

3 FEATURE ANALYSIS

In this section, we introduce how to derive the representative features, along with a systematic statistical analysis. Having an in-depth understanding of the influence of the single factor and joint influence of multiple factors can help profile team's performance and explain the effectiveness of our prediction model. Specifically, we construct two categories of features: *Static features* and *Dynamic features*.

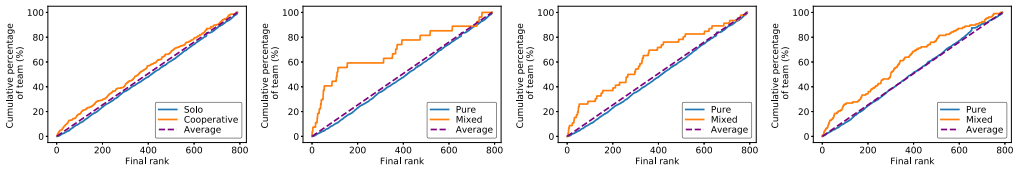
3.1 Static Features

3.1.1 Contestant Profile Features. Most contestant profile data are categorical and cannot be directly aggregated by the team. Therefore, we first extract features for each contestant. Specifically,



(a) Age v.s. rank. (b) Education level v.s. rank. (c) Industry type v.s. rank. (d) Team size v.s. rank.

Fig. 4. Static feature distributions in KDDCup.

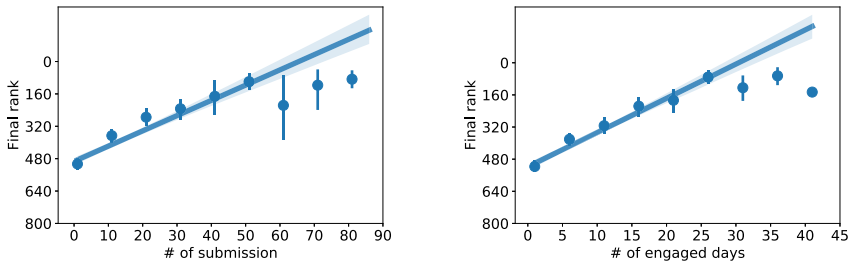


(a) Team up v.s. rank. (b) Mix company v.s. rank. (c) Mix school v.s. rank. (d) Mix industry type v.s. rank.

Fig. 5. Impact of team up in KDDCup.

we extract *age*, *gender*, *nationality*, *education level*, *job industry type*, *company*, *registration date*, *if captain*, *number of Github repositories*, *number of Github followers*, and *number of Github following*. Figure 4(a) depicts the relationship between contestants' age and final rank. We observe that the average rank of contestants in different age groups is similar, but older contestants tend to have a higher variance. Figure 4(b) shows the relationship between contestants' education level and their final rank. We observe the average rank of different education levels are similar, and the higher education level does not necessarily guarantee a higher rank. In contrast, the variance among the undergraduate students is the smallest. The relation among contestant industry type and the final rank is reported in Figure 4(c). Surprisingly, contestants majoring in IT do not dominate other contestants. In fact, we observe contestants majoring in the medical and energy industries achieve higher ranks on average.

3.1.2 Team Profile Features. We exploit *the number of team members*, *if mixed companies*, *if mixed education institutes*, *if mixed industries*, *if a merged team*, and *first submission date* as team profile features. Figure 4(d) shows the influence of team member size to final rank, with 90% confidence interval. In general, more team members tend to result in better final rank, but with higher variance. Note that over 72.8% teams in KDDCup are solo (team with only one contestant), and only 4.3% teams are with over five members, which partially explains why the final rank variance of larger teams are greater. Figure 5(a) reports the final rank distribution of solo teams and cooperative teams (teams with more than one contestant). On average, we observe cooperative teams consistently achieve higher rankings than solo teams. Looking further into the composition of cooperative teams, we discover cross-company team-up significantly helps improve the final rank of the team, as shown in Figure 5(b). We observe the similar influence of cross education institutions and cross industry type team-ups, as reported in Figure 5(c) and (d). These observations also indicate that interdisciplinary teams are generally more productive and creative [12], and can therefore achieve a higher rank.



(a) # of submission v.s. rank. (b) # of engagement days v.s. rank.

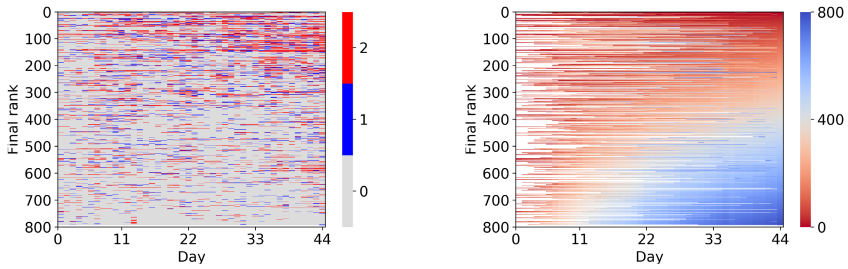
Fig. 6. Impact of team behaviors in KDDCup.

3.2 Dynamic Features

Dynamic features are those changing overtime during the competition. For most competitions, the leaderboard is updated on a daily basis. In this work, we regard one day as a basic time step and extract dynamic features for each time step.

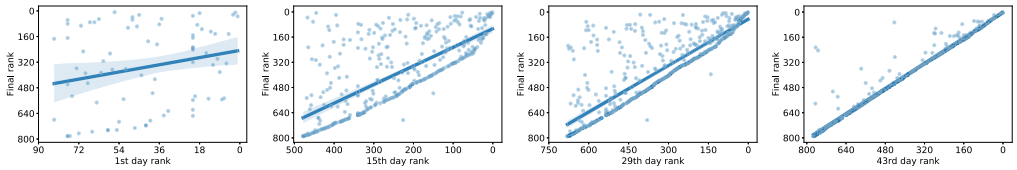
3.2.1 Team Behavior Features. For each time step, we extract corresponding *submission count*, *submission scores*, and *accumulated engagement days* (number of days have logins on the website). Besides, we further augment team behavior features including the *max submission score* in the time step, the *minimal submission score* in the time step, the *submission scores difference* in the time step, and *activeness* compared with last time step (i.e., submit more or less or equal number of results). All the above team behavior features only depend on the team itself and vary over time. We observe a positive correlation between submission count and engagement days with the final rank, as shown in Figure 6. Teams with more submissions and longer-term competition engagement tend to achieve a higher rank. Besides, the rank improvement tends to be smoother when the submission count is over a 50% max submission count limit and when the engagement days are over 50% of the competition's duration. Similar to the influence of team size, the variance also goes larger for teams with larger submission counts and longer engagement days, partially because the distribution of submission counts and engagement days also follows the long-tail distribution. Figure 7(a) further depicts team submission behaviors by day. All teams are sorted by final rank, with top-ranked teams plotted in the upper part of the figure. As can be seen, teams with higher ranks tend to be more active and submit more results during the competition. Besides, in the KDDCup competition, the top 100 teams will enter the next competition stage. We observe the submission number of top 50 teams decreasing near the competition end since they are with higher probability of entering the next stage. While teams rank between 50 and 200 tend to submit more before the competition end, fighting to enter the next stage.

3.2.2 Team Performance Features. We further extract *current rank*, *current score*, *historical highest rank*, and *total submission count* as team performance features. Different with team behavior features, team performance features are either temporally monotonic (for instance, current score is the highest historical score) or related to other teams (for example, current rank is also dependent on the rank of other teams). Figure 7(b) plots the rank of each team by day, where the red color indicates a higher rank, blue indicates a lower rank, and blank indicates no valid results yet. In Figure 7(b), teams are ordered the same as in Figure 7(a). We can make the following two observations. First, we observe "the stronger, the stronger" rule in the competition, where the leading teams tend to achieve a higher rank, and the back teams tend to lose the game as time goes on. For example, once a team falls out of the top-100, few of them can come back again. Second, top



(a) Daily submission. (b) Daily rank.

Fig. 7. Heatmaps of dynamic features in KDDCup.



(a) Final rank v.s. 1st day rank (b) Final rank v.s. 15th day rank (c) Final rank v.s. 29th day rank (d) Final rank v.s. 43rd day rank

Fig. 8. Snapshots of rank list in different competition periods v.s. final rank in the KDDCup.

ranks at the beginning do not indicate a higher final rank. In fact, as more and more teams join the competition, top teams at the beginning are evenly distributed in the final rank list. Figure 8 shows the relationship between rank list in specific days versus the final rank list. As can be seen, as the competition goes on, more and more teams join and submit their results (more data points). We also observe the rank improvement is with higher variance (points at the top left of the triangle), but the rank decline is gradual (points at the bottom right of the triangle), which can also be validated by Figure 7(b). Besides, the rank of most teams become stable as the competition goes on, which indicates the difficulty of performance improvement near the end of the competition.

3.3 Correlation Analysis

Finally, we conduct a statistical correlation analysis between the team's features and the corresponding future rank in the KDDCup. Specifically, we calculate the Pearson correlation coefficient [5] and p-value between each numerical feature and future rank, as shown in Table 3. We calculate the Pearson correlation between features and the team's final rank for static features and the feature of accumulated engaged days and total submission count. For the other dynamic features (submission count, submission scores, max submission score, minimal submission score, submission scores difference, activeness, current score, and historical highest rank), we first divide the KDDCup dataset into a series of 12-day time windows, and extract features within each time window. Then, we calculate the Pearson correlation with the team rank on the following day of the corresponding time window. Note in Table 3, a lower value of achieved rank represents a better team performance in the competition. Therefore, a negative Pearson correlation coefficient indicates the numerical feature is positively correlated with the future performance, and a smaller value indicates a higher correlation. Besides, we calculate the F-value [38] and p-value between each categorical feature and future rank, as shown in Table 4. In short, we observe static features such as registration date, number of team members, nationality, company, if caption, and team

Table 3. Pearson Correlation between Numerical Features and the Team's Future Rank

Feature	Correlation	Feature	Correlation
Age	−0.023	Accumulated engaged days	−0.524*
Register date	−0.190*	Max submission score	−0.660*
# of Github repositories	−0.105	Minimal submission score	−0.064
# of Github followers	−0.059	Submission scores difference	−0.192
# of Github following	−0.019	Activeness	−0.010
# of team member	−0.128*	Current score	−0.764*
First submission date	0.081	History highest rank	0.942*
Submission count	−0.507*	Total submission count	−0.488*
Submission scores	−0.556*		

(*indicates $p < 0.001$).

Table 4. F-value between Categorical Features and the Team's Final Rank

Feature	F-value	Feature	F-value
Gender	0.013	If captain	28.43*
Nation	3.50*	If merged team	44.20*
Education level	0.25	If mixed education institutes	10.09
Job industry type	0.94	If mixed companies	13.33*
Company	3.70*	If mixed industry types	10.23

(*indicates $p < 0.001$).

mix type are statistically significant for their future rankings. Dynamic features such as submission count, engagement days, submission scores, and current rank are also statistically significant for the future rankings. In general, we observe dynamic features have a higher correlation with the future rankings, which motivates us to devise a dedicated learning module for dynamic features in subsequent predictive analysis.

4 PREDICTIVE ANALYSIS

In this section, we present the MTRL framework for the team's future performance prediction. In the following, we use bold capital letters and bold lowercase letters to denote matrices and vectors, and use non-bold letters and squiggy letters to denote scalar variables and sets, respectively.

4.1 Formal Problem Statement

Ranking is the most representative index of the team's performance. We extract the team's future ranking as a proxy for the team's future performance. Consider a set of n teams $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ composed of m individual contestants $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$, where $n \leq m$. Let $\mathbf{x}_i^s$ denotes extracted static team profile features of team $s_i \in \mathcal{S}$, $\mathbf{x}_i^u$ denotes static contestant profile features of contestant $u_i \in \mathcal{U}$. $\mathbf{x}_i^{b,t}$ and $\mathbf{x}_i^{p,t}$ denote team behavior features and team performance features of team $s_i \in \mathcal{S}$ at time step t. The dynamic team behavior features and team performance features in previous T time steps are $\boldsymbol{\mathcal{X}}^b = \{\mathbf{X}^{b,t-T+1}, \mathbf{X}^{b,t-T+2}, \ldots, \mathbf{X}^{b,t}\}$ and $\boldsymbol{\mathcal{X}}^p = \{\mathbf{X}^{p,t-T+1}, \mathbf{X}^{p,t-T+2}, \ldots, \mathbf{X}^{p,t}\}$, respectively. We use $y_i^t \in \mathbf{y}^t$ to denote the rank of team s_i at time step t. The team's future rank prediction problem is formally defined below.

Problem 1. ***Team's future ranking prediction problem.*** *Given a historical time window T, static team profile features $\mathbf{X}^s$, contestant profile features $\mathbf{X}^u$, dynamic team behavior features $\boldsymbol{\mathcal{X}}^b$,*

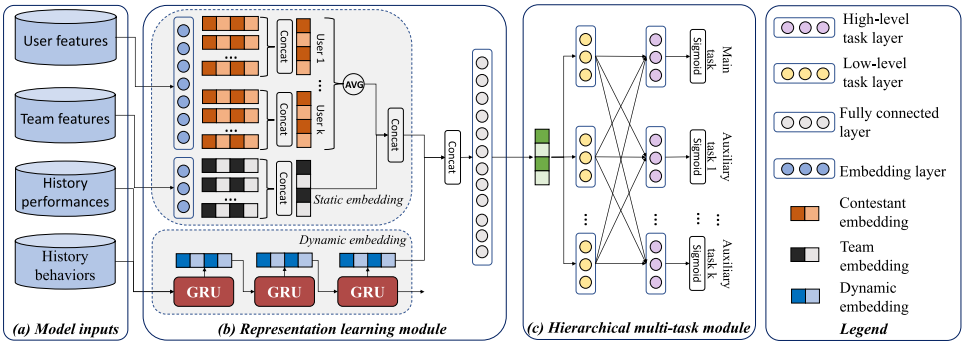


Fig. 9. Architecture overview of MTRL.

and team performance features $\mathcal{X}^p$, our problem is to forecast the rank of all $s_i \in \mathcal{S}$ over next τ time steps,

$$f(\mathbf{X}^s; \mathbf{X}^u; \mathcal{X}^b; \mathcal{X}^p) \rightarrow (\hat{\mathbf{y}}^{t+1}, \hat{\mathbf{y}}^{t+2}, \ldots, \hat{\mathbf{y}}^{t+\tau}), \tag{1}$$

where $\hat{\mathbf{y}}^{t+i}$ is the estimated rank of all teams at time step $t+i$, and $f(\cdot)$ is the mapping function we aim to learn.

4.2 Framework Overview

The architecture of MTRL is shown in Figure 9. Given features constructed in Section 3, we first propose a *representation learning* module to obtain unified representations for each team. In particular, the static embedding block applies several juxtaposed embedding layers to project sparse categorical features into low dimensional embedding vectors, and the dynamic embedding block employs a recurrent neural network to capture dependencies among dynamic features along the timeline. The overall team representation is a combination of the learned static and dynamic embedding vectors. Then, we propose a novel *hierarchical multi-task learning* module to boost the prediction performance. Specifically, the low-level task layer jointly considers the difference among team representations in multiple future time steps prediction, and the high-level task layer leverages several correlated auxiliary tasks to share task commonalities and therefore improve model generalization ability.

4.3 Learning Unified Team Representation

The team representation is a combination of static team embedding and dynamic team embedding.

Static embedding block. We leverage several distinct embedding layers [23] to project both team and contestant features into low dimensional dense vectors. Each contestant and team is represented as a concatenation of corresponding contestant embedding vectors and team embedding vectors, respectively. Since a team may contain multiple contestants, we apply an average aggregation operation to transform contestant-specific embeddings to a team level representation. Note other types of aggregation functions are also applicable for contestant embedding aggregation. The overall static representation of team s_i is derived by

$$\mathbf{h}_i^s = Embed(\mathbf{x}_i^s) \oplus \frac{1}{k_i} \sum_{j=1}^{k_i} Embed(\mathbf{x}_i^u), \tag{2}$$

where $Embed(\cdot)$ is the embedding operation, $\oplus$ is the concatenation operation, k_i is number of contestants of team s_i, and $\sum$ is a vector addition operation.

Dynamic embedding block. Dynamic features are evolving over time. Exploiting the sequential pattern and inherent correlations of dynamic features in different time steps is beneficial for the team's future rank prediction. Inspired by the recent success of **recurrent neural network** (**RNN**) on processing sequential data [22, 44], we employ the **Gated Recurrent Unit** (**GRU**) [13], a simple yet effective variant of RNN, to learn the dynamic representation. Given behavior features and performance features of previous T steps as the input, the dynamic representation of team s_i at time step t is defined as

$$\mathbf{h}_i^t = (1 - \mathbf{z}_i^t) \circ \mathbf{h}_i^{t-1} + \mathbf{z}_i^t \circ \widetilde{\mathbf{h}}_i^t, \tag{3}$$

where $\mathbf{h}_i^{t-1}$ is the dynamic representation of team s_i at last time step, $\mathbf{z}_i^t, \widetilde{\mathbf{h}}_i^t$ are defined as

$$\begin{cases} \mathbf{r}_i^t = \sigma(\mathbf{W}_r[\mathbf{h}_i^{t-1} \oplus \mathbf{x}_i^{b,t} \oplus \mathbf{x}_i^{p,t}] + \mathbf{b}_r) \\ \mathbf{z}_i^t = \sigma(\mathbf{W}_z[\mathbf{h}_i^{t-1} \oplus \mathbf{x}_i^{b,t} \oplus \mathbf{x}_i^{p,t}] + \mathbf{b}_z) \\ \widetilde{\mathbf{h}}_i^t = \tanh(\mathbf{W}_{\widetilde{h}}[\mathbf{r}_i^t \circ \mathbf{h}_i^{t-1} \oplus \mathbf{x}_i^{b,t} \oplus \mathbf{x}_i^{p,t}] + \mathbf{b}_{\widetilde{h}}) \end{cases}, \tag{4}$$

where $\boldsymbol{W}_r, \boldsymbol{W}_z, \boldsymbol{W}_{\widetilde{h}}, \boldsymbol{b}_r, \boldsymbol{b}_z, \boldsymbol{b}_{\widetilde{h}}$ are learnable parameters, $\circ$ denotes Hadamard product, and σ is the sigmoid function. The learned dynamic representation incorporates dependencies of all dynamic features in previous T steps; therefore, it is more informative for future rank prediction.

With the static representation $\mathbf{h}_i^s$ and dynamic representation $\mathbf{h}_i^d$, we can directly predict ranks of s_i in next τ time steps,

$$\left(\hat{y}_i^{t+1}, \hat{y}_i^{t+2}, \ldots, \hat{y}_i^{t+\tau}\right) = \sigma\left(\mathbf{W}_o[\mathbf{h}_i^s \oplus \mathbf{h}_i^d] + \mathbf{b}_o\right), \tag{5}$$

where $\mathbf{W}_o$ is learnable parameters and $\mathbf{b}_o$ is the bias.

4.4 Hierarchical Multi-task Learning

Since most data science competitions are only held for a short time period (typically within several months), future rank prediction suffers from the data scarcity problem. In previous studies, **Multi-task learning** (**MTL**) has been widely applied in many areas such as natural language processing [15] and image recognition [40]. It has been proven that introducing extra auxiliary tasks is beneficial to improve the prediction performance [69]. To this end, we further introduce the hierarchical multi-task learning block to share common knowledge between highly related tasks to boost the team's future rank prediction. In general, there are two types of MTL, hard parameter sharing based [9] and soft parameter sharing based [18]. We follow the hard parameter sharing paradigm [9] in the implementation, where multiple tasks share most bottom layers (i.e., the representation learning module) while having dedicated task-specific layers on top. Specifically, we propose a *hierarchical multi-task learning* based on hard parameter sharing, where different tasks share low level neural networks but have independent output layers. The intuition behind the hierarchical multi-task learning module is two-fold. First, the team performance at different future time steps is different, the team representation should incorporate the temporal difference for each future time step. However, the output layer defined in Equation (5) regards the team representation for each future time step to be identical, which neglects the temporal difference in future rank prediction. Second, as illustrated in Section 3, the future rank is highly related to many indicators, such as engagement activity and score, which can be adopted as auxiliary supervision signals. In particular, the hierarchical multi-task learning module contains two components, (1) the *low-level MTL* block where each task corresponds to a different future time step, and (2) the *high-level MTL* block where each task corresponds to a correlated signal with the future rank.

Low-level MTL. We first define the low-level MTL block to model the difference of team representations at different time steps. Specifically, given the future time step τ, we define a set of

low-level tasks as $\{\mathcal{T}^{t+1}, \mathcal{T}^{t+2}, \ldots, \mathcal{T}^{t+\tau}\}$. For each task $\mathcal{T}^{t+j}$, we aim at learning a corresponding mapping function

$$f^{t+j}\left(\mathbf{h}_i^s \oplus \mathbf{h}_i^d\right) \rightarrow \mathbf{h}_i^{t+j}, \tag{6}$$

where $\mathbf{h}_i^{t+j}$ is time-step specific representation for time step $t+j$. In this way, the low-level MTL outputs τ different team representations $\{\mathbf{h}_i^{t+1}, \mathbf{h}_i^{t+2}, \ldots, \mathbf{h}_i^{t+\tau}\}$ for team s_i. Note the low-level MTL does not have direct supervision signals. Instead, we optimize time-dependent team representations based on high-level tasks via backpropagation. Note the low-level multi-task learning distinguishes itself from traditional multi-task setting from two aspects, (1) each task in the low-level is an implicit task without direct supervision signals, and (2) the time-dependent team representations are optimized based on high-level tasks via backpropagation. Such time-dependent formulation can be analogized to positional embedding in transformer network [56], where the representation in each position should play a different role in the sequential prediction process.

High-level MTL. In the high-level MTL, except the main rank prediction task $\mathcal{T}^m$, we further introduce two auxiliary tasks, i.e., the *future score prediction* task $\mathcal{T}^{a_1}$ and the *future activity prediction* task $\mathcal{T}^{a_2}$. The two auxiliary tasks above have been proven highly correlated with the team's future rank in Section 3. Similar with the main task, the future score prediction task and the future activity prediction task aim at forecasting the score (such as accuracy, F1 score, etc.) and the activity trends of each $s_i \in \mathcal{S}$ over the next τ, respectively. In particular, we define the future activity prediction task as a multi-class classification problem, which includes three classes defined as: (1) *Improve* if the submission count is greater than yesterday, (2) *Stable* if the submission count equals to yesterday, and (3) *Decline* if the submission count is lower than yesterday. We employ a softmax function for the future activity prediction task.

One intermediate problem of hierarchical multi-task learning is the task combination explosion. Assume there are m lower-level tasks and n higher-level tasks, we need $m \times n$ task-specific layers at the high-level, which exponentially increases the learning parameters and makes the model hard to converge. To this end, we recursively reuse the high-level multi-task block for each lower-level representation. In other words, for each high-level task, we aim at learning a mapping function shared by all low-level outputs. Recursively reusing high-level tasks further shares common information among different time steps, and reduces the number of parameters of the hierarchical multi-task module from $O(mnd^2)$ to $O(md + nd)$, where d is the number of parameters in each task specific layer. For instance, for the main task $\mathcal{T}^m$, the output layer is defined as

$$\hat{y}_i^{t+j} = \sigma\left(\mathbf{W}_o^m \mathbf{h}_i^{t+j} + \mathbf{b}_o^m\right), \tag{7}$$

where $\hat{y}_i^{t+j}$ is the estimated rank of team s_i at time step $t+j$, $\mathbf{W}_o^m$ and $\mathbf{b}_o^m$ are task specific parameters. The output layer of the two auxiliary tasks are defined similarly, except we employ the softmax function for future activity prediction,

$$\hat{p}_i^{t+j} = \frac{e^{(\mathbf{W}_o \mathbf{h}_i^{t+j} + \mathbf{b}_o)}}{\sum_{k=1}^{K} e^{(\mathbf{W}_o \mathbf{h}_i^{t+j} + \mathbf{b}_o)}}, \tag{8}$$

where $\hat{p}_i^{t+j}$ is the estimated future activity probability of team s_i at time step $t+j$, K is number of classes, $\mathbf{W}_o$ and $\mathbf{b}_o$ are task specific parameters.

4.5 Optimization

In MTRL, all learnable parameters and tasks are optimized jointly. For the main task, we aim at minimizing the **mean absolute error** (**MAE**) between the predicted rank and the observed rank

$$O_1 = \frac{1}{n\tau} \sum_{i=1}^{n} \sum_{j=1}^{\tau} \left| \hat{y}_i^{t+j} - y_i^{t+j} \right|. \tag{9}$$

Similarly, the objective of the future score prediction is defined as

$$O_2 = \frac{1}{n\tau} \sum_{i=1}^{n} \sum_{j=1}^{\tau} \left| \hat{y}_i^{a_1, t+j} - y_i^{a_1, t+j} \right|, \tag{10}$$

where $\hat{y}_i^{a_1, t+j}$ and $y_i^{a_1, t+j}$ are respectively the estimated and the observed score of s_i at time step $t + j$. For the future activity prediction task, we aim at minimizing the **cross-entropy** (**CE**) loss

$$O_3 = \frac{1}{n\tau} \sum_{i=1}^{n} \sum_{j=1}^{\tau} y_i^{a_2, t+j} \log \hat{y}_i^{a_2, t+j}, \tag{11}$$

where $\hat{y}_i^{a_2, t+j}$ and $y_i^{a_2, t+j}$ are estimated and observed activity class of s_i at time step $t + j$.

Besides the objective of each task, we also incorporate the $L2$ regularization in optimization. The overall learning objective is

$$O = O_1 + \lambda_1 (O_2 + O_3) + \lambda_2 \|\mathbf{W}\|_2, \tag{12}$$

where λ_1 and λ_2 are hyper-parameters for auxiliary tasks and $L2$ regularization, respectively. Finally, we employ the Adam [29] optimizer for joint training.

5 EXPERIMENTS

5.1 Experimental Setup

We evaluate MTRL on four datasets described in Section 2. We mainly focus on: (1) the overall performance of MTRL, (2) the effectiveness of hierarchical multi-task learning, (3) the parameter sensitivity, (4) the robustness check of MTRL with different input size and output steps, (5) model generalization across different datasets, and (6) the case study. For model training, we chronologically order each dataset, and split the training set, validation set, and test set by 60%, 20%, 20%, respectively.

5.1.1 Metrics. We adopt MAE and NDCG, two widely used metrics [34, 59] to evaluate the performance. Specifically, MAE is the averaged absolute distance between the estimated rank and observed rank of each team.

$$MAE = \sum_{i=1}^{N} \frac{|\hat{y}_i - y_i|}{N}, \tag{13}$$

where $\hat{y}_i$ and y_i is the predicted rank and observed rank of team i, respectively, and N is the total number of teams.

NDCG is define as the ratio of **Discounted Cumulative Gain** (**DCG**) and **Ideal Discounted Cumulative Gain** (**IDCG**). Given a ranked team list, the DCG at position k is defined as

$$DCG(k) = \sum_{i=1}^{k} \frac{2^{rel(s_i)-1}}{\log_2(i+1)}, \tag{14}$$

Table 5. Overall Prediction Performance

Dataset	KDDCup (Mean/STD)		Billboard (Mean/STD)		RemoteSense (Mean/STD)		WSDMCup (Mean/STD)	
Algorithm	MAE	NDCG	MAE	NDCG	MAE	NDCG	MAE	NDCG
LR	36.46/2.27	0.51/0.05	21.54/1.27	0.44/0.03	19.19/0.60	0.49/0.02	4.95/0.29	0.60/0.04
RF	37.34/0.50	0.42/0.01	32.61/0.35	0.51/0.02	14.12/0.32	0.71/0.03	7.82/0.02	0.74/0.04
XGBoost	34.90/0.49	0.60/0.02	30.00/0.08	0.45/0.02	14.34/0.20	0.77/0.03	6.57/0.04	**0.84/0.04** ★
DNN	35.54/2.02	0.57/0.07	19.68/1.47	0.55/0.04	16.72/0.52	0.53/0.07	4.81/0.13	0.65/0.10
SDRNN	31.99/1.23	0.71/0.08 ★	17.20/0.58	0.59/0.03 ★	15.34/0.97	0.72/0.08 ★	4.13/0.09	0.76/0.05 ★
MTRL	**27.75/0.57**	**0.82/0.04**	**15.94/0.17**	**0.64/0.01**	**11.83/0.55**	**0.86/0.02**	**3.86/0.13**	0.83/0.05

All the improvements are statistically significant according to Welch's t-test at level 0.01 comparing MTRL to other baselines, except for results marked with "★" (Smaller MAE and larger NDCG are better).

where $rel(s_i)$ is the relevance score of the ith team. We set $rel(s_i)$ as the actual position of s_i in the ranked route list in descending order. The IDCG is defined as the ideal score DCG can achieve. Then the NDCG is defined as

$$NDCG(k) = \frac{DCG(k)}{IDCG(k)}. \tag{15}$$

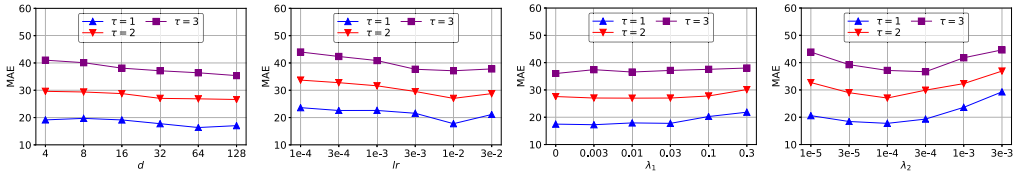
5.1.2 Implementation. Our model and all neural network based models are implemented with Pytorch. The RF and GBDT are implemented based on the XGboost library [11]. All methods are evaluated on a Linux server with 8 NVIDIA Tesla P40 GPUs. We choose the input time step $T = 12$ and $\tau = 3$ for prediction. We set the learning rate $lr = 0.01$, $\lambda_1 = 0.03$, and $\lambda_2 = 0.0001$. In each layer, we fix the hidden size to 32. For fair comparison, all parameters of each baseline are carefully tuned based on grid search strategy. The source code and data are available at https://github.com/RaymondHLIU/KDDCup-Analy.

5.1.3 Baselines. Since there are no dedicated methods for team's future rank prediction, we adopt both statistical learning and deep learning methods as baselines. Specifically, we compare our model against the following five baselines.

— **LR** uses logistic regression for future rank prediction. We concatenate static features and T step dynamic features described in Section 3 as the input. We set the learning rate $lr = 0.01$, and weight decay $\lambda_2 = 1e - 4$.
— **RF** predicts future rank via the Random Forest. The input feature is the same as LR. We set the number of trees to 150, and the maximum depth of a tree to 5.
— **XGBoost** predicts future rank using a variant of Gradient Boosting Decision Tree [11], which is also known as one of the most effective statistical learning models in many data science competitions. We set the maximum depth of a tree to 5, and learning rate $lr = 0.01$.
— **DNN** is a deep learning-based model comprising two layer fully connected hidden neural network. The input of DNN is the same as LR. The hidden size of the first and the second hidden layer is 20 and 10, respectively. We choose learning rate $lr = 0.001$, and weight decay $\lambda_2 = 1e - 3$.
— **SDRNN** employs a recurrent neural network for processing dynamic features. The architecture of SDRNN is the same as our model except for the hierarchical multi-task module. We set the learning rate $lr = 0.003$, weight decay $\lambda_2 = 1e - 4$, and fix the hidden size to 32 in each layer.

5.2 Overall Performance

Table 5 reports the overall performance of MTRL as well as all the compared baselines on four datasets with respect to MAE and NDCG. We run all methods five times with different random



(a) Effect of d. (b) Effect of lr. (c) Effect of λ_1. (d) Effect of λ_2.

Fig. 10. Parameter sensitivity on KDDCup.

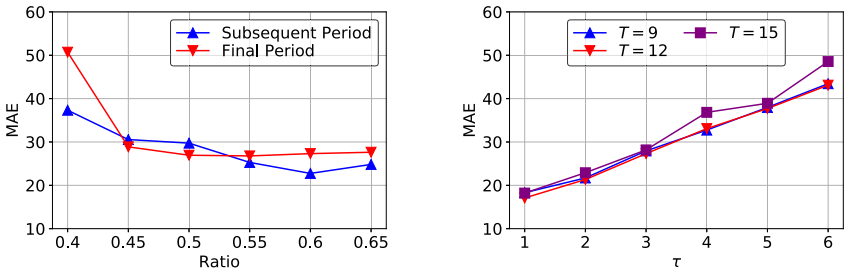
Table 6. Ablation Study of Hierarchical Multi-task Learning on KDDCup

	w/o auxiliary tasks	w/o low-level tasks	w/o score prediction	w/o trend prediction	MTRL
MAE	30	30.05	29.83	28.95	27.75
NDCG	0.71	0.8	0.76	0.72	0.82

seeds and report the mean and **standard deviation** (**STD**) of the results. Specifically, we have trained a separate model for each dataset based on the training set. The reported results are evaluated on the corresponding test set. For example, the model trained with the training set of the KDDCup is evaluated on the test set of the KDDCup only. As can be seen, our model outperforms all baselines using both MAE and NDCG on four datasets. Specifically, MTRL achieves (13.25%, 7.33%, 22.88%, 6.54%) improvement compared with the state-of-the-art baseline (SDRNN) using MAE, and the improvement using NDCG are (15.49%, 8.47%, 19.44%, 9.21%). We observe the improvement of MTRL on larger datasets are greater. The improvement of MTRL compared with SDRNN on WSDMCup is relatively small, which is possibly because it is a tiny dataset and SDRNN is powerful enough to capture useful information. We notice the results of MTRL are more statistically significant than all baselines using both metrics, except for the NDCG results of SDRNN in all datasets and XGBoost in the WSDMCup dataset. This is perhaps because we use mean square error as optimization objective rather than ranking loss, which makes the NDCG score relatively unstable. Moreover, WSDMCup is a relatively small dataset, which makes MTRL prone to overfitting. Moreover, we observe LR performs poorly on four datasets compared with other non-linear models, which demonstrates that simply applying a linear combination is not enough to fully utilize the static and dynamic features we constructed. Besides, XGBoost achieves similar or even better results compared with all deep learning-based methods except MTRL on all datasets, which demonstrate its effectiveness on capturing non-linear dependencies. Overall, the above results demonstrate the predictability of team's future performance and the effectiveness of our features as well as the predictive model.

5.3 Effectiveness of Hierarchical Multi-task Learning

Then, we evaluate the effectiveness of the hierarchical multi-task learning module to validate the benefit of each auxiliary task for the future ranking prediction. As reported in Table 6, we observe a substantial improvement on the main future ranking prediction task by introducing low-level tasks, high-level score prediction, and trend prediction tasks. Note that removing all auxiliary tasks yields slightly better MAE than only removing low-level tasks but results in higher variance (1.52 versus 0.82 standard deviations) and lower NDCG score. Moreover, we observe the **mean absolute percentage error** (**MAPE**) of the score prediction tasks on four datasets are $\{0.1, 0.12, 0.14, 0.2\}$, and the accuracy of the future activity prediction task on four datasets are $\{0.79, 0.81, 0.71, 0.79\}$, demonstrating the predictability of two auxiliary tasks.



(a) Effect of Training Set Ratio. (b) Effect of τ.

Fig. 11. Robustness check on KDDCup.

5.4 Parameter Sensitivity

Besides, we report the impact of the hidden size d, the learning rate lr, and the influence of λ_1 and λ_2 using MAE on KDDCup. The results on other datasets are similar. All parameters are tested on three future steps. Each time we vary a parameter and set others to their default values.

First, to test the impact of hidden size, we vary d from 4 to 128. The results are illustrated in Figure 10(a). As can be seen, there is a performance improvement when we increase d from 8 to 32 and a performance degradation when d is greater than 128. In fact, choosing $d = 32$ is good enough to capture the latent information from features. A larger d will induce extra computation overhead and make the model hard to converge.

Next, we vary lr from $1e-4$ to $3e-2$. The results are depicted in Figure 10(b). We observe that our model yields an optimal performance when $lr = 1e-2$, and a performance degradation when we further increase or decrease lr. The performance of MTRL is relatively stable when lr is smaller than $1e-2$, and a larger learning rate induces suboptimal prediction results.

To test the impact of auxiliary tasks, we vary λ_1 from 0 to 0.3. The results are reported in Figure 10(c). MTRL yields an optimal performance when $\lambda_1 = 0.003$ and is not very sensitive to λ_1. To test the impact of L2 regularization, we vary λ_2 from $1e-5$ to $3e-3$. As reported in Figure 10(d), we observe the best performance when $\lambda_2 = 1e-4$. Overall, our model is more sensitive to λ_2, and a relatively small λ_2 is good enough for regularization.

5.5 Robustness Check

After that, we evaluate the robustness of MTRL with (1) different training set ratio, and (2) different input time step T and output time step τ. First, we vary the training test ratio from 0.4 to 0.65, test the model on subsequent 20% data and the latest 20% data, respectively. As shown in Figure 11(a), we observe a better prediction performance when we increase the ratio of training set. Besides, we observe consistently larger predictive error in the final period. This makes sense because the uncertainty of the team's rank goes large in the distant future. Overall, the team's performance is also predictable by using early period features as training data, but with a relatively larger error. Then, we vary τ from 1 to 6, and vary T from 9 to 15 . As shown in Figure 11(b), the predictive error linearly increases when we increase the prediction time step. This makes sense because the uncertainty of the team's rank goes large in the distant future. Moreover, we observe that the MAE fluctuates slightly when $T = 15$. This is because of the duration of competitions usually short, too long input sequence further reduces the number of training instances, and may induce unnecessary noise.

Table 7. Performance on Different Target Datasets

Source \ Target	KDDCup		Billboard		RemoteSense		WSDMCup	
	MAE	↑ Δ	MAE	↑ Δ	MAE	↑ Δ	MAE	↑ Δ
KDDCup	27.31	0	14.85	−5.71%	8.25	−26.54%	2.57	−31.10%
Billboard	36.63	34.13%	15.75	0	9.3	−17.19%	2.88	−22.79%
RemoteSense	45.83	67.81%	24.24	53.90%	11.23	0	4.13	10.72%
WSDMCup	49.19	80.12%	25.15	59.68%	11.48	2.23%	3.73	0

Table 8. Result of Top-10 Teams on KDDCup

Team name	Predicted rank	Final rank
π	1	1
Stacking Artascope	2	4
β-kdd	3	2
Tuosanqianxumi	4	5
DeepBlueAI	5	8
Wangxiaohu327	6	11
Tiyuyitiaojie	7	6
Dongfangshenghao	8	9
Yuanshanwenjiayoulongyu	9	10
Magic Click	10	14

5.6 Model Generalization

To test the generalization of our proposed model, we save the parameters of the best MTRL models trained on the training set of KDDCup, Billboard, RemoteSense, and WSDMCup, respectively. The hyperparameters of MTRL models are the same as those reported in Table 5. Then, we evaluate each model on the test set of these four datasets to assess the generalization ability of the model. We measure the MAE on each dataset and calculate the relative difference of MAE, which are shown in Table 7. Overall, we observe a better generalization ability of models trained on larger datasets. Compared with the model trained on Billboard, RemoteSense, and WSDMCup, the model trained on KDDCup achieves 5.71%, 26.54%, and 31.1% prediction reduction on Billboard, RemoteSense, and WSDMCup test set, respectively. On the other hand, the model trained on smaller datasets achieves a worse prediction performance on larger datasets, for example, model trained on RemoteSense and test on KDDCup result in 67.81% larger prediction error. The above results validate our assumption that data scarcity is a major bottleneck in team's future rank prediction. Besides, the above results suggest knowledge sharing methods such as transfer learning is perhaps an important direction in which to further improve the prediction performance. We left it as the future work.

5.7 Case Study

To further understand the prediction results of our model, Table 8 reports the predicted top-10 teams by MTRL and their actual final ranking in KDDCup. Overall, we observe the predicted rank matches or nearly matches the final ranks for most teams except the 6th team. By further look into the data, we find the score of the 6th team remains unchanged in the last few days and surpassed by teams in the behind with significant improvements (two teams achieved over 50 rank improvement in last four days). This result indicates modeling such sudden changes in a short period can further improve the overall prediction performance.

6 DISCUSSION

In this section, we discuss the key findings and implications of this work and point out several limitations.

Findings. Our quantitative and predictive analysis reveal several facts. First, we find online data science competitions successfully attract worldwide contestants of different backgrounds, such as age, nation, education level, and industry. Second, we identify both static and dynamic features such as team size, submission count, and daily rank are positively correlated with the team's future performance with statistical significance and are helpful for the team's future rank prediction. In contrast, features such as education level and Github activity, which are thought to be significant professional indicators, do not have a strong correlation with the team's future performance. Third, we observe "the stronger, the stronger" rule in data science competitions, where front teams tend to keep their advantage and dominate back teams in the rest of the competition. Fourth, the higher-ranked teams tend to be more positive in the competition, and the rank list tends to be stable as time goes on. Finally, we proved the team's future performance is predictable based on our constructed features and MTRL.

Implications. Based on the above findings and our experience of hosting the KDD Cup 2019 machine learning competition, for future competitions, we have several tips for organizers:

— *Control the duration.* A moderate length competition is enough to distinguish expertise from the crowd, and obtain a good enough solution, control the duration to trade-off the expected solution quality and operation costs.
— *Launch promotion events in the middle.* we recommend organizers to launch promotion events such as release a better baseline in the middle of the competition to improve the engagement of middle back contestants.

Besides, we also have several tips for contestants:

— *Form a diverse team.* Forming a team with members of diversified backgrounds such as industry type, company, and education institution sustainability improves the contestant's performance. Actually, even simply forming a team also has a positive effect on the final ranking, meaning that "Two heads are better than one".
— *Try more submissions.* Teams with more submissions tend to achieve a higher final rank. In fact, trying more submissions not only verifies the score of different solutions, but also helps disentangle interference of different features and model components, which in consequence provides deeper insights into improving the performance.
— *Keep engaged.* Keeping engaged opens a window to leverage the intelligence from other contestants. Activities such as forum discussions and tracking new baselines are also helpful for improving the overall performance.

In the future, we envision our study helping contestants for competition strategy design, and organizers for proactive competition management.

Limitations. Our work is limited by the data that we have access to. All competitions used in this work were hosted on the Dianshi platform. To ensure the above findings and implications are generalizable and not site-specific, future research will need to test on other data science competition platforms such as Kaggle and Tianchi. Besides, the feature analysis conducted in this work mainly focuses on correlation analysis. The audience should keep in mind that representative indicators do not necessarily mean causation of the team's future ranking. In addition, we only involve Github profile data for data augmentation. The proposed framework can be further improved by extending more data sources. We anticipate future work will integrate more contestant information such as Kaggle profile and contestant discussion record to improve the model effectiveness.

7 RELATED WORK

Our work is highly related to *crowd-sourcing based problem solving*, *team profiling and performance prediction*, *sequence prediction,* and *multi-task deep learning*.

Crowd-sourcing based problem solving. Data science competition is directly related to crowd-sourcing based problem solving, where the form can be either a contest or a collaboration. This process can be either one-shot or long-lasting, and the expected output ranges from a machine learning model [4], a software [2, 7], and a mathematical problem solution [54], to even a logo [63]. The majority of existing studies mainly focus on empirical study of contest mechanisms and the user behavior impact. To name a few, Athanasopoulos et al. [3] find real-time updated leaderboard is helpful for improving the accuracy of the final result, Hill et al. [26] conclude that too few or too many collaborators negatively influence the quality of the final solution, Tausczik et al. [53] investigate the influence of open sharing on the quality of the final solution. Moreover, Archak et al. [2] find contestants are strategically choosing competitions to maximize their reward, Wang et al. [57] study factors that influence contestant sustained participation, and Bullinger et al. [8] find the positive effect of team cooperation, which also validates our observations in data science competitions. In this work, we focus on the data science competition and conduct a comprehensive quantitative and predictive analysis to understand the team's time-evolving performance in depth.

Team profiling and performance prediction. There have been works on characterizing the factors that could influence team performance. For example, pobiedina et al. [50] suggest that the cooperation and social ties within a team are crucial factors for the success of a team. Moreover, the national diversity of a team has the potential to influence performance. Nascimento et al. [47] figure out the difference in team cooperation strategies and members' skills between experienced and inexperienced teams. Cheng et al. [12] confirm the positive impact of team diversity on performance and the team behaviors during the competition. Ye et al. [65] further emphasize the importance of demographics of the individual member, geographic information between team members, and the role of being the captain in influencing the member performance during the competition.

The team performance prediction has been applied in multiple scenarios, such as education [10, 21, 36, 49], sports [32, 45], and esports [27, 46]. Previous studies generally model the team performance based on the static team and user profile [32, 49], dynamic team performance and behaviors [10, 21, 36, 46, 60], or the combination of two types of features [27, 45]. For instance, Omar et al. [49] propose a rough sets approach in predicting the collaboration efficiency of students based on the team profile. Müller et al. [46] predict the team performance in online games by using the attributes of the dynamic communication network with the random forest model. Min et al. [45] forecast the pairwise winning probability in football matches by combining both the team profile and runtime status using a Bayesian method. Hodge et al. [27] conduct real-time team winning prediction in a video game with a set of machine learning models, such as logistic regression and tree-based models. This work investigates the team rank prediction in the context of the real-world data science competition, which is seldom discussed in the literature. Compared with previous studies, we incorporate both the users' and teams' static features and the dynamic in-competition features with a deep learning approach. Meanwhile, we adopt the strategy of multi-task learning in our prediction framework.

Sequence prediction. Deep learning has been widely adopted in various sequence prediction tasks, such as user behavior prediction, spatiotemporal prediction, and financial prediction. For online user behavior prediction, Liu et al. [37] predict user's future engagement time in a social app by leveraging their historical behavior via a deep sequential model. Miao et al. [43] predict topic trends and popularity in microblogs in an online fashion. Eldele et al. [19] recognize human activity by learning the time-series representation in an unsupervised fashion. In sequential recommendation,

each user is represented as a sequence of interacting items and a series of most likely interacting items in the future are generated [51, 61, 68]. In spatiotemporal prediction tasks, both the temporal autocorrelation of each sequence and the spatial correlation among sequences are utilized for forecasting. For instance, Liang et al. [34] predict air quality by modeling the mutual impact of signal sequences, and Han et al. [24] jointly forecast air quality and weather conditions based on graph neural network and adversarial training. Wang et al. [58] solve the problem of traffic flow prediction through aggregating information from adjacent roads. Zhang et al. [66] predict parking availability by combining the information of nearby parking lots. For financial prediction, Li et al. [31] predict the ever-changing stock movement via tensor model, and Liu et al. [20] jointly predict stock trend and prices via a temporal graph neural network. In this work, we leverage the deep recurrent neural network to capture the sequential behaviors of each team to predict the teams' future performance.

Multi-task deep learning. On one hand, deep learning has been widely used in many areas such as computer vision and natural language processing, because of its effectiveness on learning and generalizing feature representation [1, 30, 33, 41, 62]. On the other hand, multi-task learning has been proven beneficial for boosting model performance by transferring common knowledge among tasks [69]. Based on the information sharing method, multi-task deep learning can be categorized into hard parameter sharing based [9, 67] and soft parameter sharing based [18, 42]. In general, multi-task deep learning is adopted with the parameter sharing approach [70], and we do likewise. Recent advances of multi-task learning focus on dynamically modeling task relationships [64, 70], and some recent studies [25, 52] has successfully facilitated multiple tasks in lower neural network layers to guide low-level representation learning. Inspired by the above studies, we propose a novel hierarchical multi-task representation learning model for team's future rank prediction, where the low-level team representations are jointly supervised by high-level tasks. To the best of our knowledge, this is the first attempt to apply multi-task deep learning to help understand data science competitions.

8 CONCLUSION

In this article, we investigated the time-evolving team performance in data science competitions. We identified and quantified four categories of representative features that are significantly correlated with team's future performance. We further proposed a multi-task deep learning-based model, MTRL, for team's future rank prediction. In particular, we introduced a representation learning module to project and aggregate both static and dynamic features into a unified embedding vector. Moreover, we proposed a hierarchical multi-task learning module to capture the inner-connection among time-dependent team representations as well as multiple relevant auxiliary tasks. Extensive experimental results on three real-world data science competitions demonstrate the predictability of the team's future performance by using our constructed features and MTRL. Our analysis and prediction framework has been deployed on the Baidu–Dianshi platform to instrument and forecast team performance. In the future, we plan to investigate the causal relationships among features, and explore contestant behaviors and performances across multiple data science competitions to improve the generalizability of MTRL.

REFERENCES

[1] Faizan Ahmad, Ahmed Abbasi, Jingjing Li, David G. Dobolyi, Richard G. Netemeyer, Gari D. Clifford, and Hsinchun Chen. 2020. A deep learning architecture for psychometric natural language processing. *ACM Transactions on Information Systems* 38, 1, Article 6 (Feb. 2020), 29 pages.

[2] Nikolay Archak. 2010. Money, glory and cheap talk: Analyzing strategic behavior of contestants in simultaneous crowdsourcing contests on TopCoder. com. In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 21–30.

[3] George Athanasopoulos and Rob J. Hyndman. 2011. The value of feedback in forecasting competitions. *International Journal of Forecasting* 27, 3 (2011), 845–849.
[4] Robert M. Bell, Yehuda Koren, and Chris Volinsky. 2010. All together now: A perspective on the netflix prize. *Chance* 23, 1 (2010), 24–29.
[5] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Proceedings of the Noise Reduction in Speech Processing*. 1–4.
[6] Kevin J. Boudreau and Karim R. Lakhani. 2013. Using the crowd as an innovation partner. *Harvard Business Review* 91, 4 (2013), 60–69.
[7] Luka Bradeško, Michael Witbrock, Janez Starc, Zala Herga, Marko Grobelnik, and Dunja Mladenić. 2017. Curious cat–mobile, context-aware conversational crowdsourcing knowledge acquisition. *ACM Transactions on Information Systems* 35, 4, Article 33 (Aug. 2017), 46 pages.
[8] Angelika C. Bullinger, Anne-Katrin Neyer, Matthias Rass, and Kathrin M. Moeslein. 2010. Community-based innovation contests: Where competition meets cooperation. *Creativity and Innovation Management* 19, 3 (2010), 290–303.
[9] R. Caruana. 1997. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning (ICML'93)*. 41–48.
[10] Ling Cen, Dymitr Ruta, Leigh Powell, Benjamin Hirsch, and Jason Ng. 2016. Quantitative approach to collaborative learning: Performance prediction, individual assessment, and group composition. *International Journal of Computer-Supported Collaborative Learning* 11, 2 (2016), 187–225.
[11] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 785–794.
[12] Ziqiang Cheng, Yang Yang, Chenhao Tan, Denny Cheng, Alex Cheng, and Yueting Zhuang. 2019. What makes a good team? A large-scale study on the effect of team composition in honor of kings. In *Proceedings of the World Wide Web Conference*. 2666–2672.
[13] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proceedings of the NIPS 2014 Workshop on Deep Learning, December 2014*.
[14] Aaron Clauset, Cosma Rohilla Shalizi, and Mark E. J. Newman. 2009. Power-law distributions in empirical data. *SIAM Review* 51, 4 (2009), 661–703.
[15] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*. 160–167.
[16] Chrysanthos Dellarocas. 2003. The digitization of word of mouth: Promise and challenges of online feedback mechanisms. *Management Science* 49, 10 (2003), 1407–1424.
[17] Baidu Dianshi. 2020. Retrieved 1 Feb., 2022 from https://dianshi.bce.baidu.com/competition.
[18] Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. 845–850.
[19] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. 2021. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*. 2352–2359.
[20] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019. Temporal relational ranking for stock prediction. *ACM Transactions on Information Systems* 37, 2, Article 27 (March 2019), 30 pages.
[21] F. Giannakas, C. Troussas, I. Voyiatzis, and C. Sgouropoulou. 2021. A deep learning classification framework for early prediction of team-based academic performance. *Applied Soft Computing* 106, 3 (2021), 107355.
[22] Klaus Greff, Rupesh K. Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. 2016. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems* 28, 10 (2016), 2222–2232.
[23] Huifeng Guo, Ruiming TANG, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A factorization-machine based neural network for CTR prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 1725–1731.
[24] Jindong Han, Hao Liu, Hengshu Zhu, Hui Xiong, and Dejing Dou. 2021. Joint air quality and weather prediction based on multi-adversarial spatiotemporal networks. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*.
[25] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1923–1933.
[26] Benjamin Mako Hill and Andrés Monroy-Hernández. 2013. The remixing dilemma: The trade-off between generativity and originality. *American Behavioral Scientist* 57, 5 (2013), 643–663.
[27] Victoria J. Hodge, Sam Michael Devlin, Nicholas John Sephton, Florian Oliver Block, Peter Ivan Cowling, and Anders Drachen. 2019. Win prediction in multi-player esports: Live professional match prediction. *IEEE Transactions on Games* 13, 4 (2019), 368–379.

[28] Kaggle. 2020. Retrieved 1 Feb., 2022 from https://www.kaggle.com/.
[29] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the ICLR (Poster)*.
[30] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436.
[31] Qing Li, Yuanzhu Chen, Li Ling Jiang, Ping Li, and Hsinchun Chen. 2016. A tensor-based information framework for predicting the stock market. *ACM Transactions on Information Systems* 34, 2, Article 11 (Feb. 2016), 30 pages.
[32] Yongjun Li, Lizheng Wang, and Feng Li. 2021. A data-driven prediction approach for sports team performance and its application to national basketball association. *Omega* 98 (2021), 102123.
[33] Bin Liang, Chenwei Lou, Xiang Li, Lin Gui, Min Yang, and Ruifeng Xu. 2021. Multi-modal sarcasm detection with interactive in-modal and cross-modal graphs. In *Proceedings of the 29th ACM International Conference on Multimedia*. 4707–4715.
[34] Yuxuan Liang, Songyu Ke, Junbo Zhang, Xiuwen Yi, and Yu Zheng. 2018. GeoMAN: Multi-level attention networks for geo-sensory time series prediction. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 3428–3434.
[35] Hao Liu, Yongxin Tong, Panpan Zhang, Xinjiang Lu, Jianguo Duan, and Hui Xiong. 2019. Hydra: A personalized and context-aware multi-modal transportation recommendation system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2314–2324.
[36] Sa Liu, Lin Wang, Sien Lin, Zhi Yang, and Xiaofan Wang. 2017. Analysis and prediction of team performance based on interaction networks. In *Proceedings of the 2017 36th Chinese Control Conference*. IEEE, 11250–11255.
[37] Yozen Liu, Xiaolin Shi, Lucas Pierce, and Xiang Ren. 2019. Characterizing and forecasting user engagement with in-app action graph: A case study of snapchat. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2023–2031.
[38] Richard G. Lomax. 2000. *Statistical Concepts: A Second Course for Education and the Behavioral Sciences*. Routledge.
[39] Kai Lu, Wenjun Zhou, and Xuehua Wang. 2014. Social network of the competing crowd. In *Proceedings of the 2014 International Conference on Behavioral, Economic, and Socio-Cultural Computing*. IEEE, 1–7.
[40] Xiaoqiang Lu, Xuelong Li, and Lichao Mou. 2014. Semi-supervised multitask learning for scene recognition. *IEEE Transactions on Cybernetics* 45, 9 (2014), 1967–1976.
[41] Michal Lukasik, Kalina Bontcheva, Trevor Cohn, Arkaitz Zubiaga, Maria Liakata, and Rob Procter. 2019. Gaussian processes for rumour stance classification in social media. *ACM Transactions on Information Systems* 37, 2, Article 20 (Feb. 2019), 24 pages.
[42] Rui Mao and Xiao Li. 2021. Bridging towers of multi-task learning with a gating mechanism for aspect-based sentiment analysis and sequential metaphor identification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 13534–13542.
[43] Zhongchen Miao, Kai Chen, Yi Fang, Jianhua He, Yi Zhou, Wenjun Zhang, and Hongyuan Zha. 2016. Cost-effective online trending topic detection and popularity prediction in microblogging. *ACM Transactions on Information Systems* 35, 3, Article 18 (Dec. 2016), 36 pages.
[44] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association* . International Speech Communication Association, 1045–1048.
[45] Byungho Min, Jinhyuck Kim, Chongyoun Choe, Hyeonsang Eom, and R. I. Bob McKay. 2008. A compound framework for sports results prediction: A football case study. *Knowledge-Based Systems* 21, 7 (2008), 551–562.
[46] Siegfried Müller, Raji Ghawi, and Jürgen Pfeffer. 2020. Using communication networks to predict team performance in massively multiplayer online games. In *Proceedings of the 2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE, 353–360.
[47] Fernando Felix do Nascimento Junior, Allan Sales da Costa Melo, Igor Barbosa da Costa, and Leandro Balby Marinho. 2017. Profiling successful team behaviors in league of legends. In *Proceedings of the 23rd Brazillian Symposium on Multimedia and the Web*. 261–268.
[48] Intellectual Property Office. 2019. *Artificial intelligence: A worldwide overview of AI patents and patenting by the UK AI sector*.
[49] Mazni Omar, Sharifah-Lailee Syed-Abdullah, and Naimah Mohd Hussin. 2011. Developing a team performance prediction model: A rough sets approach. In *Proceedings of the International Conference on Informatics Engineering and Information Science*. Springer, 691–705.
[50] Nataliia Pobiedina, Julia Neidhardt, Maria del Carmen Calatrava Moreno, and Hannes Werthner. 2013. Ranking factors of team success. In *Proceedings of the 22nd International Conference on World Wide Web*. 1185–1194.
[51] Ruihong Qiu, Zi Huang, Jingjing Li, and Hongzhi Yin. 2020. Exploiting cross-session information for session-based recommendation with graph neural networks. *ACM Transactions on Information Systems* 38, 3, Article 22 (May 2020), 23 pages.

[52] Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 231–235.
[53] Yla Tausczik and Ping Wang. 2017. To share, or not to share? Community-level collaboration in open innovation contests. *Proceedings of the ACM on Human-Computer Interaction* 1, CSCW (2017), 100.
[54] Yla R. Tausczik, Aniket Kittur, and Robert E. Kraut. 2014. Collaborative problem solving: A study of mathoverflow. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 355–367.
[55] Tianchi. 2020. Retrieved from https://tianchi.aliyun.com/home/.
[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems*. 5998–6008.
[57] Xuan Wang, Hanieh Javadi Khasraghi, and Helmut Schneider. 2019. What sustains individuals' participation in crowdsourcing contests?. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*.
[58] Xiaoyang Wang, Yao Ma, Yiqi Wang, Wei Jin, Xin Wang, Jiliang Tang, Caiyan Jia, and Jian Yu. 2020. Traffic flow prediction via spatial temporal graph neural network. In *Proceedings of the Web Conference 2020*. Association for Computing Machinery, New York, NY, 1082–1092.
[59] Yining Wang, Liwei Wang, Yuanzhi Li, and Di He. 2013. A theoretical analysis of NDCG ranking measures. In *Proceedings of the COLT*.
[60] Oryza Wisesa, Andi Adriansyah, and Osamah Ibrahim Khalaf. 2020. Prediction analysis sales for corporate services telecommunications company using gradient boost algorithm. In *Proceedings of the 2020 2nd International Conference on Broadband Communications, Wireless Sensors and Powering*. IEEE, 101–106.
[61] Feng Xue, Xiangnan He, Xiang Wang, Jiandong Xu, Kai Liu, and Richang Hong. 2019. Deep item-based collaborative filtering for top-n recommendation. *ACM Transactions on Information Systems* 37, 3, Article 33 (April 2019), 25 pages.
[62] Su Yan and Xiaojun Wan. 2015. Deep dependency substructure-based learning for multidocument summarization. *ACM Transactions on Information Systems* 34, 1, Article 3 (July 2015), 24 pages.
[63] Yang Yang, Pei-yu Chen, and Rajiv Banker. 2011. Winner determination of open innovation contests in online markets. In *Proceedings of the ICIS*. 1–16.
[64] Yaqiang Yao, Jie Cao, and Huanhuan Chen. 2019. Robust task grouping with representative tasks for clustered multi-task learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1408–1417.
[65] Teng Ye, Wei Ai, Lingyu Zhang, Ning Luo, Lulu Zhang, Jieping Ye, and Qiaozhu Mei. 2020. Predicting individual treatment effects of large-scale team competitions in a ride-sharing economy. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2368–2377.
[66] Weijia Zhang, Hao Liu, Yanchi Liu, Jingbo Zhou, and Hui Xiong. 2020. Semi-supervised hierarchical recurrent graph neural network for city-wide parking availability prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 1186–1193.
[67] Weijia Zhang, Hao Liu, Lijun Zha, Hengshu Zhu, Ji Liu, Dejing Dou, and Hui Xiong. 2021. MugRep: A multi-task hierarchical graph representation learning framework for real estate appraisal. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 3937–3947.
[68] Yuan Zhang, Fei Sun, Xiaoyong Yang, Chen Xu, Wenwu Ou, and Yan Zhang. 2020. Graph-based regularization on embedding layers for recommendation. *ACM Transactions on Information Systems* 39, 1, Article 2 (Sept. 2020), 27 pages.
[69] Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering* (2021), 1–1. DOI: 10.1109/TKDE.2021.3070203
[70] Jiejie Zhao, Bowen Du, Leilei Sun, Fuzhen Zhuang, Weifeng Lv, and Hui Xiong. 2019. Multiple relational attention network for multi-task learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1123–1131.

Received March 2021; revised November 2021; accepted January 2022