Mention Recommendation in Twitter with Cooperative Multi-Agent Reinforcement Learning

Tao Gui*, Peng Liu*, Qi Zhang†, Liang Zhu, Minlong Peng, Yunhua Zhou and Xuanjing Huang
School of Computer Science, Shanghai Key Laboratory of Intelligent Information Processing
Fudan University, Shanghai, P.R.China 201203
{tgui16,pengliu17,liangzhu17,mlpeng16,zhouyh16,xjhuang}@fudan.edu.cn

ABSTRACT

In Twitter-like social networking services, the "@" symbol can be used with the tweet to mention users whom the user wants to alert regarding the message. An automatic suggestion to the user of a small list of candidate names can improve communication efficiency. Previous work usually used several most recent tweets or randomly select historical tweets to make an inference about this preferred list of names. However, because there are too many historical tweets by users and a wide variety of content types, the use of several tweets cannot guarantee the desired results. In this work, we propose the use of a novel cooperative multi-agent approach to mention recommendation, which incorporates dozens of more historical tweets than earlier approaches. The proposed method can effectively select a small set of historical tweets and cooperatively extract relevant indicator tweets from both the user and mentioned users. Experimental results demonstrate that the proposed method outperforms state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → *Collaborative search*; • **Human-centered computing** → *Social recommendation.*

KEYWORDS

Social Medias; Mention Recommendation; Reinforcement Learning

ACM Reference Format:
Tao Gui*, Peng Liu*, Qi Zhang†, Liang Zhu, Minlong Peng, Yunhua Zhou and Xuanjing Huang. 2019. Mention Recommendation in Twitter with Cooperative Multi-Agent Reinforcement Learning. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, Article 4, 10 pages. https://doi.org/10.1145/3331184.3331237

1 INTRODUCTION

Twitter speaks its own language and one of its well-known aspects is the "@mention" function. A mention is a tweet containing the "@" symbol followed by a Twitter handle[1]. Inserting mentions into a tweet serves to grab the attention of corresponding users. The mention utility allows users to spread information far beyond their own neighborhoods and improve its visibility by making it available to an appropriate set of users. Moreover, as mentions are listed in a separate tab, they attract greater attention than regular posts [25]. According to a 2018 Twitter statistic[2], the average Twitter user has 707 followers. Hence, it would be beneficial for Twitter to automatically recommend to users a small number of candidates when they want to mention others in a certain tweet.

The task of mention recommendation is to predict who should be mentioned for a given new tweet. Currently, the solution adopted by social media platforms like Twitter, Weibo, and Facebook is to provide mention suggestions after a user inputs the mention symbol "@." The suggestions given by these services are usually based either on the completion of partial inputs or users' mention histories, which tend to be less than ideal [32]. Recently, many researchers have used various Twitter information sources to facilitate recommendations, including the user's own tweet history, his/her retweet history, and the social relations between users [1, 15, 29, 39]. Gong et al. [8] take into consideration the content of the tweet and the

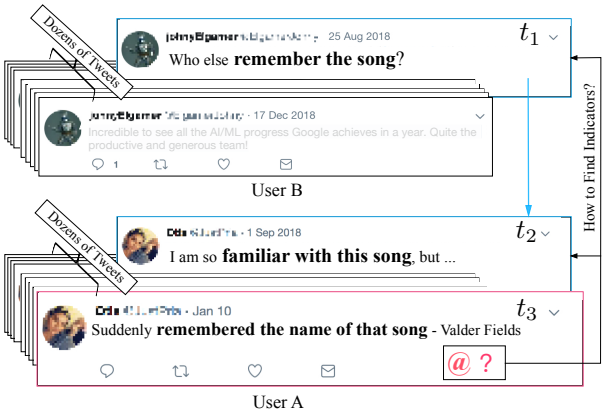

Figure 1: Examples of historical tweets posted by users and their mentioned users. To make a correct mention recommendation for the tweet "Suddenly remembered the name of that song," the key is to find indicator tweets (in blue) from the earlier tweet interactions of dozens of historical tweets.

*Both authors contributed equally to this research.
†Corresponding author.

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331237

[1]https://help.twitter.com/en/using-twitter/types-of-tweets
[2]https://www.websitehostingrating.com/twitter-statistics/

histories of candidate users via a translation process that maps the content of tweets to usernames. Huang et al. [10] use a memory network to incorporate the content of a tweet, the history of its author, and the interests of candidate users. In addition, additional visual information has been incorporated into a cross-attention memory network to perform the mention recommendation task [18, 33]. All these methods aim to measure the similarity between the interests of candidate users and the given tweet.

Although previous approaches incorporated various types of information in the modeling of mention behaviors have achieved great success, they still have difficulty in dealing with a large number of historical user tweets. Because a user may receive too many status updated from his/her followers and many tweets do not necessarily reflect the information needs of users [1, 22]. Thus, the use of several recent tweets or the random selection of a few tweets will miss important information and introduce a lot of irrelevant information, which can negatively impact the mention recommendation. As shown in Figure 1, both the tweet histories of a user and those of candidate mentioned users are useful in making a mention recommendation for the tweet, "Suddenly remembered the name of that song." The key is to find the indicator tweets (in blue) among all the earlier tweet interactions in the tweet histories of the user and candidate mentioned users. Because of the complicated relationship of communications among users [27], it is difficult to cooperatively select relevant tweets from dozens of historical tweets. Previous approaches take just a few historical tweets into consideration. For instance, Huang et al. [10] restricted the capacity of their model to five tweets randomly selected from the user's tweet history. These kinds of methods limit the capacity of modeling mention behaviors.

To tackle the above challenges, in this work, we propose the use of a novel cooperative multi-agent approach to mention recommendation. The proposed method uses two policy gradient agents to simultaneously select a small set of historical tweets from the user and candidate mentioned users and evaluates the utility of joint actions based on the classification accuracy. In cooperative settings, joint selection typically generates only global rewards, which make it difficult to determine the contribution of each selector to the group's success [5]. To overcome this problem, the proposed model employs novel cooperative reverse operation multi-agent (CROMA) policy gradients. The CROMA takes an actor-critic [12] approach characterized by differentiated advantages, in which each actor (i.e., user tweet selector or candidate-mentioned-user tweet selector) is trained by following its own unique gradient estimated by a critic. To do so, we adopt a centralized training framework with decentralized execution, in which the critic is only used during learning, and the actor is only needed during execution. In addition, we use a reverse operation to obtain an advantage for each agent. We provide each agent with a shaped reward that compares the current global reward to the reward received when the agent takes a reverse action (operation). Experimental results show that the proposed method can perform much better than existing state-of-the-art methods.

The main contributions of this work can be summarized as follows:

- We propose a method for considering a large number of historical tweets in the mention recommendation task. The use of more historical tweets enables better modeling of user mention behaviors and provides more valuable information.
- To achieve this task, our approach utilizes a novel multi-agent reinforcement learning method, CROMA, in which the user tweet selector and candidate-mentioned-user tweet selector cooperatively extract indicator content.
- To evaluate the proposed approach, we constructed a dataset containing 50 historical tweets per user [3]. Experimental results from the benchmark test indicate that the performance of CROMA is significantly better than those of other baseline approaches.

2 RELATED WORK AND BACKGROUND

This work is related to two research threads: social media user recommendation and reinforcement learning (RL). In this section, we first summarize related work and then give a brief description of the differences between existing approaches and our own. Then, we give some background on single-agent RL, including the deep Q-Network, policy gradient, and actor-critic approaches. We also discuss the settings used in multi-agent RL and the roles of reward shaping in multi-agent settings.

2.1 Mention Recommendation

As Twitter has grown into one of the most popular microblogging services, much research has focused on the analysis of the personal interests of users and building corresponding recommender systems, such as content recommendation [1, 13], community recommendation [17, 38], hashtag recommendation [2, 7, 14], and mention recommendation [8, 10, 31].

The mention recommendation task has been studied from various perspectives. Wang et al. [31] formulated this task as a ranking problem and adopted features including the user interest match, content-dependent user relationship, and user influence to make a recommendation according to the diffusion-based relevance of newly defined information. Li et al. [15] considered mention recommendation to be a probabilistic problem and utilized a probabilistic factor graph model to identify the user having the maximal capability and potential to facilitate tweet diffusion. Tang et al. [29] employed a ranking support vector machine (SVM) model to recommend the top K appropriate users to mention. The above methods require the extraction of a variety of features, including content-, social-, location- and time-based features. Gong et al. [8] proposed a topical translation model that incorporates the content of tweets and users' post histories to deal with this problem.

Recently, deep-learning-based methods have opened up new possibilities for mention recommendation. Huang et al. [10] adopted a memory network with a hierarchical attention mechanism for this task. Ma et al. [18], Wang et al. [33] proposed the incorporation of visual and textual information by a cross-attention memory network to perform mention recommendation. However, these methods only consider short-term user posting histories or a random sample of a few historical tweets, which are insufficient for modeling user interest. In this work, we use a novel cooperative multi-agent method to model long-term user posting histories and effectively select relevant historical tweets.

[3] https://github.com/mritma/CROMA

2.2 Single-Agent Reinforcement Learning

In a single-agent, fully observable RL setting [26], an agent observes the current state $s_t \in S$ in discrete time steps t, chooses an action $a_t \in A$ according to a potentially stochastic policy π, observes a reward signal r_t, and transitions to a new state s_{t+1}. The objective in optimizing the policy parameters, θ, is to maximize the expected reward by performing actions drawn from the policy π.

Q-Learning and Deep Q-Networks. In particular environments, the Q-learning technique [34] estimates the Q-values (or action-values) $Q^\pi(s,a)$, which is a scalar used to estimate the expected sum of the gamma-discounted rewards that will accrue by taking action a in state s by following policy π as $Q^\pi(s,a) = \mathbb{E}[R|s_t = s, a_t = a]$. This Q function can be rewritten by applying the *Bellman equation* to the immediate reward and the Q-values of the next state and action as $Q^\pi(s,a) = \mathbb{E}_{s'}[r(s,a) + \gamma \mathbb{E}_{a' \in \pi}[Q^\pi(s',a')]]$. Deep Q-Networks [19] extend standard Q-learning by the use of a deep neural network as a Q-value function approximator and minimizing the following loss function:

$$\mathcal{L} = \mathbb{E}_{s,a,r,s'}[(Q^*(s,a|\theta) - y)^2], \tag{1}$$

where $y = r_t + \gamma \max_{a'} \bar{Q}(s', a'|\bar{\theta})$ is the update target given by the target network $\bar{Q}$, whose parameters $\bar{\theta}$ are periodically updated with the most recent θ value to stabilize the learning [20].

Policy Gradient and Actor-Critic Algorithms. Policy gradient methods, another type of RL technique, rely upon optimizing parameterized policies to maximize the expected return $J(\theta) = \mathbb{E}_\pi[R]$ by gradient ascent. These methods do not suffer from many of the problems that plague Q-learning methods, such as the lack of guaranteed value function, the intractability issue associated with uncertain state information, and the complexity arising from continuous states actions. A classic example is the REINFORCE algorithm [35], in which the gradient is as follows:

$$\nabla J(\theta) = \mathbb{E}_\pi[\sum_{t=0}^{T} \nabla_\theta [\log \pi_\theta(a_t|s_t;\theta) R]]. \tag{2}$$

Alternatively, using the Q function as a critic for estimating rewards has led to a class of *actor-critic* algorithms, e.g., that use the *temporal difference* (TD) error $r_t + \gamma V(s_{t+1} - V(s_t))$ [26].

2.3 Multi-Agent Reinforcement Learning

In this work, we consider multi-agent domains that are fully cooperative and partially observable, all of which are attempting to maximize the discounted sum of the joint rewards and no single agent can observe the state of the environment. Multi-agent RL can be described as an extension of Markov decision processes (MDPs). The MDPs for N agents can be described as a stochastic game G, represented as a tuple $G = \langle N, S, A, \{R_i\}_{i \in N}, \mathcal{T} \rangle$, where S is the set of states; A is the collection of action sets, with a^i being i-th agent's action; $\mathcal{T}$ is the state transition function: $\mathcal{T} : S \times A \to S$; and $\{R_i\}_{i \in N}$ is the set of reward functions. By their joint actions, each agent receives a reward for judging its own action $r_i : S \times A \to \mathbb{R}$, and aims to maximize its total expected return $R_i = \sum_{l=0}^{T} \gamma^l r_{t+l}$, where γ is the discount factor and T is the total number of time steps.

Multi-Agent Using Team Rewards. The use of Q-Networks has been extended to cooperative multi-agent settings, in which each agent learns its own Q-function that is conditional only on the state and its own action. This represents the simplest and most popular approach to multi-agent RL. Independent Q-learning (IQL) is implemented by basing the optimized action-observation history of each agent on the same team rewards [28]. IQL is useful because it avoids the scalability problems associated with trying to learn a joint Q-function that relies on a joint action space A, which grows exponentially with the number of agents. IQL is also naturally suited to partially observable settings, since, by construction, it learns decentralized policies in which each agent's action is based only on its own observations.

Multi-Agent Using Shaped Rewards. Although IQL is very successful, it introduces a key problem: the environment becomes nonstationary from the point of view of each agent, as it contains other agents who are themselves learning, which rules out any convergence guarantees [6]. In addition, it is difficult to determine the contribution of each agent to the group's success [5]. Hence, a shaped reward must be allocated to each agent according to its own contribution to the team rewards. The idea is inspired by difference rewards [4, 30], in which each agent learns from a shaped reward that compares the global reward to the reward to that received when that agent's action is replaced with a default action. However, access to a simulator or estimated reward function is required, and in general it is unclear how to choose the default action.

Therefore, a core issue in multi-agent learning is how to design a reward for each agent, because joint actions typically generate only global rewards. The actions of all the agents contribute to that global reward, which can make the gradient of each agent very noisy [5]. In the next section, we describe in detail our proposed novel multi-agent RL technique for tackling the above problems.

3 APPROACH

Given a query tweet t_q issued by an author v, we must recommend one or several users u from a list of candidate mentioned users U according to their long-term historical tweets H. Because there are too many historical user tweets and variety of content types, we must extract many of the relevant indicator tweets from the historical tweets of the user and candidate mentioned users.

Figure 2 shows the architecture of our proposed model, which has three components. First, we use a tweet encoder to represent tweets, and historical tweets are encoded using a hierarchical attention model. Second, we propose the use of CROMA policy gradients to select relevant indicator tweets, which adopt a centralized training framework with decentralized execution in which a centralized critic and differentiated advantages are applied. Both the user and candidate-mentioned-user tweet selectors are policy gradient agents, which take the query tweet t_q and historical tweets H as inputs and determine which historical tweets should be selected. The selectors are trained by following the different gradients estimated by the critic. The differentiated advantages are shaped rewards that compare the current global reward to those received when each agent's action is replaced with a reverse action. Finally, we merge the query-tweet representation and the features selected by agents, and use a fully connected softmax layer for prediction.

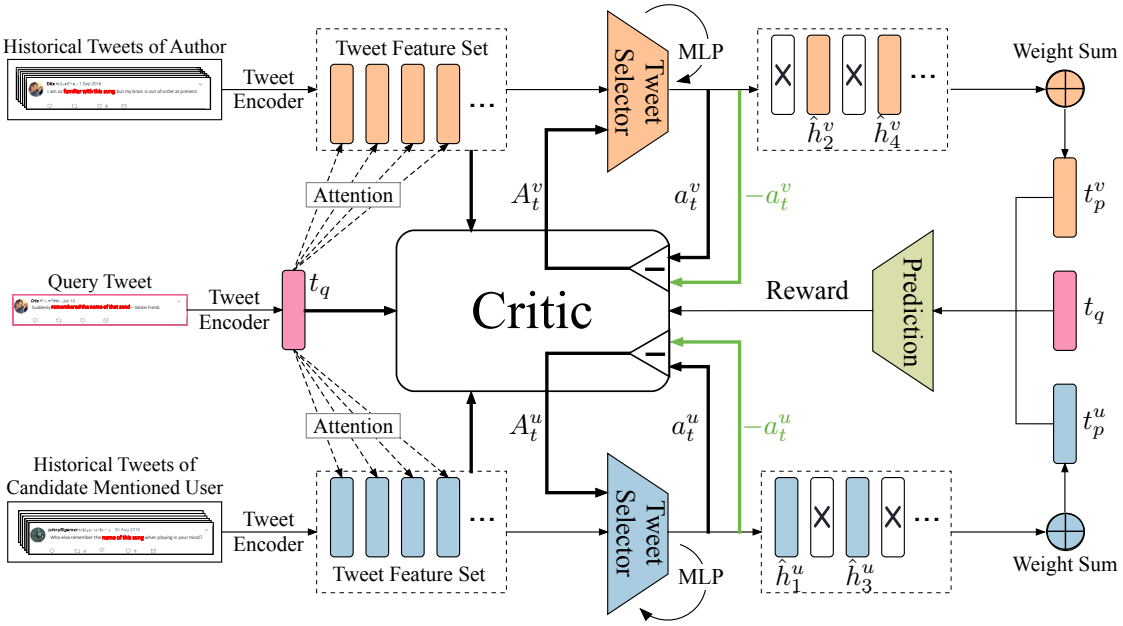

Figure 2: Architecture of the proposed model. At each time step t, the advantage A_t^e of selector e is given by comparing the current global reward to the reward received when that agent's action is replaced with an opposite action $-a_t^e$.

3.1 Tweet Encoder

We denote the query tweet of the author as t_q, historical tweets of the author as $D^v = \{t_1^v, t_2^v, \cdots, t_N^v\}$, and historical tweets of a candidate mentioned user as $D^u = \{t_1^u, t_2^u, \cdots, t_N^u\}$, where N is the number of historical tweets. We assume that each tweet consists of a sequence of words $t_i = \{w_1, w_2, \cdots, w_M\}$, where $t_i \in D$ and M is the length of a tweet. We use a tweet encoder similar with Huang et al. [10] to obtain query-tweet representations and historical tweet representations.

Query-Tweet Representation

We use a simple but effective method to represent the query tweet t_q, in which a tweet consists of a bag of words $\{w_1, w_2, \cdots, w_M\}$, where each word is a one-hot vector. We use a randomly initialized word embedding matrix $A \in \mathbb{R}^{d \times |V|}$ to store the representations of all words, where d is the embedding dimension and $|V|$ is the size of the vocabulary. Thus, each word w_i in t_q is embedded in a continuous space. We then sum these embedding vectors to obtain a representation of the query tweet: $t_q = \sum_i Aw_i$. The query tweet representation t_q is also used to obtain historical tweet representations.

Historical Tweet Representations

Historical tweets are a set of texts posted by a user, with each tweet consisting of a bag of words. Because each tweet has different importance to the query tweet and not all the words in tweets are equally important, we introduce a hierarchical-attention method to model historical tweets on both word and tweet level. Taking the author of a query tweet as an example, the historical tweet representations are formulated as follows.

Word-Level Attention. For each tweet $t_i \in D$, we obtain a word embedding matrix representation $W = [Aw_1, Aw_2, \cdots, Aw_M]$ by looking up the word vocabulary table A. Then, we compute the attention weights through the inner product between word representation $Aw_j \in W$ and query tweet representation t_q with a softmax function as follows:

$$Pr_{ij} = \frac{\exp(t_q^T Aw_j)}{\sum_{m=1}^{M} \exp(t_q^T Aw_m)}, \tag{3}$$

where Pr is the probability over the input position.

Then, we sum all the word representations using the word-level attention weights to obtain the tweet-level representation:

$$h_i = \sum_j Pr_{ij} Aw_j \tag{4}$$

where h_i is the embedding of the i-th tweet $t_i \in D$. We use a simplified function to represent the word-level attention: $h_i = WAtt(t_i, t_p)$.

Using the above procedure, all historical tweets are transformed into vectors of the same dimension, which are then used to model the user long-term mention behaviors.

Tweet-Level Attention. In the next section, we introduce methods for selecting the relevant tweet history set. For simplicity, we first use all of the user tweet histories to match the query tweet, although this brings a lot of irrelevant information (which could negatively impact the mention-recommendation outcome). To aggregate the tweet interests, we use a tweet-level attention mechanism to allocate different weights to each historical tweet. Given the embedding set of tweets $H = \{h_1, h_2, \cdots, h_N\}$, the interest representation of the historical tweets is the weighted sum of these

embedded tweets as follows:

$$\begin{aligned} m_{h_i} &= \tanh\left(W_o t_q + W_h h_i\right) \\ Pr_{h_i} &= \frac{\exp(W_{mh}^T m_{h_i})}{\sum_{n=1}^{N} \exp\left(W_{mh}^T m_{h_n}\right)} \\ t_p &= \sum_{i}^{N} Pr_{h_i} h_i, \end{aligned} \tag{5}$$

where t_p is the representation of the user's historical tweets. W_o, W_h, and W_{mh} are the parameters of the attention functions. Pr_{h_i} is the weight divided by the input tweets, which can be interpreted as the degree of importance of a particular tweet in the tweet set. We use a simplified function to represent tweet-level attention: $t_p^v = TAtt(H^v, t_q)$, $t_p^u = TAtt(H^u, t_q)$.

As described above, we obtain the author's query-tweet representation t_q, the author's tweet-history representation t_p^v, and the candidate-mentioned-user's tweet-history representation t_p^u. Inspired by [10], we use multi-hop function to aggregate them and obtain the output, as follows:

$$\begin{aligned} O_1 &= t_q + TAtt(H^v, t_q) + TAtt(H^u, t_q) \\ O_k &= O_{k-1} + TAtt(H^v, O_{k-1}) + TAtt(H^u, O_{k-1}), \end{aligned} \tag{6}$$

where $k \in 1, 2, \cdots, K$ is the number of hops, and O_K is the final output for prediction. We find that using a multi-hop mechanism in tweet-level attention refines the tweet-history representation to improve the matching performance.

3.2 Tweet Selectors with Independent Q-learning

When we wish to select a relevant query-tweet subset from historical tweets, a key challenge arises when we encounter the diverse content of posts and their multi-faceted points of interest. In addition, the nature of discrete selection decisions makes losses no longer differentiable. To overcome this problem, we introduce a multi-agent actor-critic method, independent Q-learning (IQL), to select a relevant query-tweet subset for the user and candidate mentioned users.

A certain user's historical tweets H correspond to the sequential inputs of one episode. For each user selector, at each step $t \in T$, the agent chooses an action a_t (selecting the current tweet or not) after observing the state s_t. When all of the selections are made, the matching predictor will give a delayed reward to update the parameters of agent θ_a. The goal is to learn a policy π_{θ_a} that, given an observed state s_t, estimates a distribution for the next action a_t, which is denoted by $\pi_{\theta_a}(a_t|s_t)$. The environment is determined by the initial state s_1 and transition function $s_{t+1} = f(s_t, a_t)$. We consider the agent to use the standard reinforcement learning framework, in which the agent participates in a Markov decision process (MDP). Then, the probability of an action sequence $a_{1:T}$ in an episode can be expressed as follows:

$$\pi(a_{1:T}) = \prod_{t=1}^{T} \pi(a_t|s_t),\ s_{t+1} = f(s_t, a_t). \tag{7}$$

At the end of the episode, the delayed reward r provided by the environment is used to optimize θ_a such that the agent can adopt a better policy.

Next, we introduce several key points with respect to the agent, including the state representation s_t, the action a_t, and the reward function. The historical tweet selectors of the author and candidate mentioned users have the same structure.

State representation. Assume that we have obtained tweet representations by the tweet encoder. At step t, the model has obtained t historical tweets as inputs, which are denoted as $H_{1:t}$. Given $H_{1:t}$, the policy gradient agent could make the following observations: the current post representation h_t, the indicator post set $H_{indi} = [\hat{h}_1, \hat{h}_2, ...]$, and the irrelevant post set $H_{irre} = [\check{h}_1, \check{h}_2, ...]$. The notations $\hat{h}$ and $\check{h}$ will be defined in the **action** step. Note that at the initial time, H_{indi} and H_{irre} are empty sets, and we use zero vectors to initialize these two sets. We thereby formulate the agent's state s_t as follows:

$$s_t = [t_q \otimes h_t \otimes O_K \otimes avg(H_{indi}) \otimes avg(H_{irre})], \tag{8}$$

where avg indicates the average pooling operation and $\otimes$ the concatenation operation.

Action. By sampling from the multinomial distribution, the agent takes an action a_t at step t using policy $a_t \sim \pi(s_t, a_t; \theta_a)$. We define action $a_t \in \{1, 0\}$ to indicate whether the agent will select the current tweet h_t. Therefore, we can adopt a logistic function to sample actions from the policy function as follows:

$$\begin{aligned} \pi(a_t|s_t; \theta_a) &= Pr(a_t|s_t) \\ &= a_t * \sigma(MLP(s_t)) \\ &\quad + (1 - a_t) * (1 - \sigma(MLP(s_t))), \end{aligned} \tag{9}$$

where MLP represents the multilayer perceptron used to map the state s_t to a scalar, and $\sigma(\cdot)$ is the sigmoid function. If the agent takes an action to select the tweet ($a_t = 1$), then the hidden state h_t will be rewritten as $\hat{h}$ and appended in H_{indi}. Otherwise, it will be rewritten as $\check{h}$ and appended in H_{irre}.

Reward function. After executing a series of actions, the agent will construct a query-tweet-indicator representation set H_{indi}. We set the reward to be the likelihood of ground truth after finishing all the selections made for the i-th user. In addition, to encourage the model to delete more tweets, we include a regularization factor to limit the number of selected posts as follows:

$$r_i = Pr(y_i|H_{indi}; \theta_d) - \lambda T'/T, \tag{10}$$

where T' refers to the number of selected tweets and λ refers to a hyperparameter used to balance the reward. By setting the reward to be the likelihood of ground truth, we capture the intuition that optimal selections will increase the probability of ground truth. Therefore, by interacting with the classifier via rewards, the agent is incentivized to select optimal tweets from H for training a better classifier.

3.3 Cooperative Multi-Agent Selecting Algorithm

Because of the complicated interaction that characterizes the communication of users [27], we must jointly consider the user and candidate mentioned users to select relevant historical tweets. Hence, the policy gradient agents should not be independent of each other. We propose the use of novel cooperative multi-agent reinforcement

learning (CROMA) to select relevant tweets from the tweet histories of both the user and candidate mentioned users.

Centralized Critic

At each step t in one episode, we obtain a subset containing the selected features for predicting the mention action. The selected features of the user and candidate mentioned users are denoted by H^v_{init} and H^u_{init}, respectively. Then the tweet feature representation function 6 can be rewritten as follows:

$$\begin{aligned} O^t_1 &= t_q + TAtt(H^v_{init}, t_q) + TAtt(H^u_{init}, t_q) \\ O^t_k &= O^t_{k-1} + TAtt(H^v_{init}, O^t_{k-1}) + TAtt(H^u_{init}, O^t_{k-1}), \end{aligned} \tag{11}$$

where k and t refers to the number of iterations and time steps, respectively.

The matching problem is a universal binary classification problem. As such, we process this representation using a dropout operation for two fully connected layers (i.e., MLP). The output of the last layer is followed by a sigmoid non-linear layer that predicts the probability distribution for the two classes:

$$Pr(y_t = \hat{y}|O^t_K; \theta_d) = \hat{y}\sigma(O^t_K) + (1-\hat{y})(1-\sigma(O^t_K)), \tag{12}$$

where $\hat{y}^t$ represents the prediction probabilities at time step t using H^v_{init} and H^u_{init}. To encourage the selector to take better actions, we can relate the reward to the likelihood of ground truth $Pr(y = \hat{y})$. A simple way to do so is to use the actor-critic algorithm, which applies the change in $Pr(y = \hat{y})$ after updating its sets with the newly chosen examples as the unified TD error, as reported in [36]:

$$\begin{aligned} r_t &= Pr(y_{t+1} = \hat{y}|O^{t+1}_K) - Pr(y_t = \hat{y}|O^t_K) \\ \mathcal{L}_t(\theta_c) &= [r_t + \gamma Q(\mathbf{H}^{t+1}_{init}, \Pi_{t+1}, \mathbf{a}_{t+1}) - Q'(\mathbf{H}^t_{init}, \Pi_t, \mathbf{a}_t)]^2, \end{aligned} \tag{13}$$

where $\mathbf{H}^t_{init} = H^v_{init} \otimes H^u_{init}$, $\Pi = \pi^v \otimes \pi^u$,, and $\mathbf{a} = a^v \otimes a^u$. $\otimes$ is the concatenation operation. However, using the same advantages makes it difficult to identify the contribution of each selector. Hence, differentiating these advantages poses a key challenge.

Differentiated Advantages Using Reverse Operation

In fact, a centralized critic can be used to implement difference rewards in our CROMA setting. Although CROMA learns a centralized critic, which estimates Q-values for joint action $\mathbf{a}$ depending on the central state $\mathbf{H}^t_{init}$, we can attribute a particular advantage to each agent by comparing the global reward to the reward assigned when an agent takes a reverse action. Intuitively, this mechanism can give a positive reward when the Q-value of a gold action subtracts that of a reverse action. Formally, for each selector $e \in \{u, v\}$, we can then compute an advantage function that compares the Q-value of the current action a^e to a misoperation baseline that takes a reverse action $-a^e$, while keeping fixed the other agent's action a^{-e}:

$$\begin{aligned} A^e(\mathbf{H}^t_{init}, \Pi_t, \mathbf{a}_t) =& Q(\mathbf{H}^t_{init}, \Pi_t, (a^e_t, a^{-e}_t)) \\ & - Q(\mathbf{H}^t_{init}, \Pi_t, (-a^e_t, a^{-e}_t)). \end{aligned} \tag{14}$$

Hence, $A^e(\mathbf{H}^t_{init}, \Pi_t, \mathbf{a}_t)$ computes a separate baseline for each agent and a centralized critic to consider each advantage. Therefore, the model can be optimized according to Algorithm 1.

3.4 Prediction

Based on the selected historical-tweet and query-tweet representations obtained from the above process, we introduce an MLP and a softmax layer to determine whether or not a certain candidate mentioned user u should be recommended for the author's "@" action in the tweet t_q. The feature representation is passed into the full-connection hidden layer:

$$f = \sigma(W_m O^T_K + b_m), \tag{15}$$

where W_m is the weight vector of the hidden layer, b_m is the bias, O^T_K is the final representation obtained from the last hop and last time step, and $\sigma(\cdot)$ is the non-linear activation function.

Finally, we use a softmax layer for our prediction:

$$p(y = \hat{y}|f; \theta_s) = \frac{\exp(\theta^i_s) f}{\sum_j \exp(\theta^j_s f)} \tag{16}$$

According to the scores output from the softmax layer, we can list the top-ranked recommended users for the "@" action.

Algorithm 1 CROMA for Mention Recommendation

1: Randomly initialize critic network $Q(S, \pi, \mathbf{a}|\theta_Q)$ and two selectors $\pi(s|\theta^e_\pi)$ with weights θ_Q and θ^e_π.
2: Initialize target networks Q' and π' with weights $\theta_{Q'} \leftarrow \theta_Q$, $\theta^e_{\pi'} \leftarrow \theta^e_\pi$. Initialize replay buffer R
3: **for** episode = 1, M **do**
4: Receive initial observation state h^e_1
5: **for** $t = 1, T$ **do**
6: Select action $a^e_t = \pi(h^e_t|\theta^e_\pi)$ according to current policy
7: Execute action a^e_t and observe the likelihood of ground truth $\Pr(y = \hat{y}_u|O_t)$ and observe the new state h^e_{t+1}
8: Execute action a^e_{t+1} and observe the likelihood of ground truth $\Pr(y = \hat{y}_u|O_{t+1})$, thereby obtain the reward $r_t = \Pr(y = \hat{y}_u|O_{t+1}) - \Pr(y = \hat{y}_u|O_t)$
9: Store transition $(H^t_{init}, \mathbf{a}_t, r_t, H^{t+1}_{init})$ in R
10: Sample a random minibatch of N transitions $(H^i_{init}, \mathbf{a}_i, r_i, H^{i+1}_{init})$ from R
11: Set $z_i = r_i + \gamma Q'(H^{i+1}_{init}, \Pi_{i+1}, \mathbf{a}_{i+1})$
12: Update critic by minimizing the loss:
$$\mathcal{L}(\theta_Q) = \frac{1}{N}\sum_i [z_i - Q(H^i_{init}, \Pi_i, \mathbf{a}_i|\theta_Q)]^2$$
13: Update selectors using differentiated advantages:
$$A^e(H, \Pi, \mathbf{a}) = Q(H, \Pi, \mathbf{a}) - Q(H, \Pi, (-a^e, a^{-e}))$$
$$\nabla_{\theta^e_\pi} J(\theta^e_\pi) = \nabla_{\theta^e_\pi} \log \pi(a^e_t|h^e_t) A^e(H, \Pi, \mathbf{a})$$
14: Update the target networks:
$$\theta_{Q'} = \tau\theta_Q + (1-\tau)\theta_{Q'}, \theta^e_{\pi'} = \tau\theta^e_\pi + (1-\tau)\theta^e_{\pi'}$$
15: **end for**
16: Update the mention classifier by minimizing the cross entropy loss:
$$J(\theta_C) = -[y \log \hat{y} + (1-y)\log(1-\hat{y})]$$
17: **end for**

4 EXPERIMENTS

In this section, we first describe the datasets and then the baseline methods we used, including a number of classic methods and a series of neural networks methods. Finally, we detail the configuration of the proposed model.

Table 1: Statistics of the Constructed Dataset, where # represents the number of tokens in datasets.

# Central Users	2,801
# Query tweets	106,384
# Candidate Mentioned Users	11,086
# Historical Tweets per User	50
# Total Historical Tweets	694,350
# Avg.Query Tweets per Author	37.98
# Avg.Mentioned Users per Author	40.01

4.1 Datasets

To evaluate the effectiveness of our proposed model, we expanded the existing dataset [10] from five tweets per user to 50 tweets per user from Twitter, which is 10 times larger than before.

We used the dataset constructed in [10], which contains 3,150 central authors and a total of 133,267 query tweets with at least one "@username." However, in this dataset, each user contains just five historical tweets. So, we expanded the dataset from five historical tweets per user to 50, and removed users having less than 50 historical tweets. Thus, we obtained 2,801 central authors with mention behavior, and a total of 106,384 query tweets. The number of mentioned users was 11,086. Finally, we crawled 50 historical tweets from each user, and collected 694,350 historical tweets. From the statistics shown in Table 1, we can see that the average number of query tweets per central author was 37.98, and the average number of users that the central author mentioned was 40.01. For each query tweet, we treated the list of mentioned users annotated by authors as the ground truth, and the user mention history of each author as candidates. We split the dataset into training, development, and testing sets in an 80/10/10 ratio.

We conducted experiments on the above datasets, and found that the historical tweets selected by our model greatly improved the performance of the mention recommendation task.

4.2 Comparison Methods

Next, as baselines for comparison, we applied several classic and state-of-the-art methods, including feature-based and neural networks methods.

Feature-based methods. As reported in previous work [10, 18], we also evaluated several classic feature-based methods on the proposed dataset, including the following:

- **NB**: Naive Bayes [24] is implemented using bag-of-word features transformed from the posting history.
- **PMPR**: The personalized mention probabilistic ranking system (PMPR) is proposed in [15] to solve the mention recommendation problem.
- **CAR**: The context-aware at recommendation (CAR) model is a ranking support vector machine model proposed in [29] to locate target users.

Neural network methods. We compared our model performance with those of some neural network methods that use the tweet histories of both authors and candidate mention users to identify mention actions.

- **LSTM**: LSTM [9] is a general method for modeling text. We used LSTM to model tweets and make predictions.
- **Attention methods**: We applied several recently proposed attention methods to the task of mention recommendation, including multi-level attention (MLAN) [37], co-attention (CAN) [16], dual-attention (DAN) [23], and modality attention (MAN) [21] methods. These methods are designed to model the relationship between objects from different perspectives.
- **AU-HMNN**: AU-HMNN is proposed in [10], which incorporates the textual information of query tweets and user histories. This is a state-of-the-art approach to the mention recommendation task.
- **Independent Q-Learning:** A strong baseline model uses selectors with same structure, and unified advantages are then generated to optimize the selectors [3].
- **Random sampling:** We also randomly sampled half of the tweets from each user to train the baseline model.

4.3 Initialization and Hyperparameters

In this work, we first restrict the number of historical tweets to 50, and then test the performance of different size of historical tweets. The maximum length of each tweet is 32. The word embeddings and other parameters related to the deep learning models by randomly sampling from a standard normal distribution and a uniform distribution in [-0.05, 0.05], respectively. We set the dimensionality of the word embedding to 300 and the number of hops to 6. The learning rate was set to 0.01, and the dropout rate was set to 0.2. Each of selectors use a MLP with 50 hidden neurons.

Our model can be trained end-to-end using back propagation, and we performed gradient-based optimization using the Adam update rule [11], with a learning rate of 0.001.

5 RESULTS AND ANALYSIS

In this section, we detail the performances of the proposed and baseline models, and present the results of a series of experiments, which demonstrate the effectiveness of the proposed model from different aspects.

5.1 Method Comparison

We compared the mention recommendation performance of the proposed and baseline models in terms of four selected measures, i.e., precision, recall, F1, and mean reciprocal rank (MRR). We also used Hits@3 and Hits@5 to represent the percentage of correct results recommended from the top N results. Table 2 shows a summary of our results. We can see that our proposed model (CROMA) performs better than the other comparison methods.

In the table, Category I shows a comparison of the feature-based methods, which use various features for training. We can see that

Table 2: Comparison results on the testing dataset. We divided the compared approaches into three categories based on different mechanisms. Category I belongs to traditional machine learning methods. Category II is based on deep neural networks. Category III includes different variants of our approach.

	Method	Precision	Recall	F-Score	MRR	Hits@3	Hits@5
I	**NB**, Pedregosa et al. 2011 [24]	51.42	50.37	50.89	63.09	67.09	78.73
	PMPR, Li et al. 2011 [24]	58.10	57.39	57.74	69.85	73.42	86.36
	CAR, Tang et al. 2015 [29]	59.74	58.62	59.17	70.57	74.68	87.34
II	**LSTM**, Hochreiter and Schmidhuber 1997 [9]	65.54	64.60	65.07	74.53	78.48	90.31
	CAN, Lu et al. 2016 [16]	63.29	62.66	62.97	71.38	76.52	90.58
	MLAN, Yu et al. 2017 [37]	60.16	59.53	59.84	71.37	77.22	91.14
	DAN, Nam et al. 2017 [23]	73.42	72.78	73.10	80.94	82.28	91.37
	MAN, Moon et al. 2018 [21]	68.35	67.72	68.03	75.18	77.22	88.61
	AU-HMNN, Huang et al. 2017 [10]	74.23	73.05	73.64	81.16	83.54	92.41
III	**Random Sampling**	70.94	69.72	70.32	77.70	82.88	93.67
	IQL, Tampuu et al. 2017 [28]	71.04	70.26	70.65	79.01	82.13	92.16
	CROMA	**74.55**	**74.09**	**74.32**	**81.85**	**86.36**	**95.00**

Table 3: Comparison of different methods between adding CROMA RL and without CROMA RL for F1, Hit@3, and Hit@5 scores. Results annotated with * are obtained when the number of historical tweets per user is restricted to five, others are trained with all 50 historical tweets.

Method	F1		Hit@3		Hit@5		
	w/o RL	w/ RL	w/o RL	w/ RL	w/o RL	w/o RL *	w/ RL
LSTM	65.07	**+0.87**	78.48	**+1.33**	90.31	-0.44	**+0.95**
CAN	62.97	**+1.23**	76.52	**+1.83**	90.58	-1.85	**+0.39**
MLAN	59.84	**+1.05**	77.22	**+1.62**	91.14	-0.81	**+1.17**
DAN	73.10	**+0.71**	82.28	**+0.86**	91.37	+1.04	**+1.20**
MAN	68.03	**+0.94**	77.22	**+1.91**	88.61	-6.33	**+1.81**
AU-HMNN	73.64	**+0.68**	83.54	**+2.82**	92.41	+0.00	**+2.59**

CAR performed better, which indicates that SVM-based methods would be effective in mention recommendation task.

Category II shows the neural networks-based methods, which avoid complicated feature design. LSTM is commonly used in text modeling and can achieve an F1 score greater than 65%, which indicates the power of neural networks-based methods for this task. We also tested five more complex classification methods on our dataset, including state-of-the-art methods for visual question-answering tasks (CAN, DAN and MLAN), named entity recognition (MAN), and mention recommendation (AU-HMNN).These five methods use different attention mechanism to model the relationship between entities. We can see that although these methods apply more complex attention strategy, some of them are not better than the simple LSTM model, such as CAN and MLAN. Due to the use of too many historical tweets, attention-based methods have difficulty focusing on the indicator tweets, which would be detailed in Section 5.3. The method AU-HMNN [10] uses a memory network to store historical tweets, and achieves start-of-the-art results. However, like other attention-based methods, the AU-HMNN can hardly avoid the interference of irrelevant tweets, which limits it's performance. Hence, the CROMA, which can effectively handle noisy historical tweets, achieves better results.

Category III shows the proposed methods adopting different strategies in selecting historical tweets. We set Random Sampling method to randomly select half of the historical tweets. Compared to random sampling models, the model that adopts unified advantages (IQL) achieves better performance, with an 70.65% F1 score, but still not better than the state-of-the-art mention recommendation method (AU-HMNN). As explained above, when using unified advantages it is difficult to optimize each selector. If we use the differentiated advantages proposed in our model to optimize the selectors, the CROMA achieves the best performance, with a value of 74.32% for the F1 measure. The Hits@3 and Hits@5 results, in particular, are respectively 86.36% and 95.00%, almost 3% better than the state-of-the-art methods. In addition, the proposed method obtains the highest MRR score, which indicates that the ground-truth mentioned users in our recommendation system obtain higher average recommendation rankings. Therefore, using differentiated advantages is much more effective than using unified advantages.

In Figure 3, we plot the precision, recall, and F1 score of the different methods with various numbers of recommended users. The number of recommended users ranges from one to five. Based on the results, we can see that the performance of CROMA is the highest of all the curves. In particular, when the number of recommended users is set to five, the proposed method obtains a value of 95% for recall, which is much better than the other methods. Next, we demonstrate why our selection strategy is more effective than other strategies.

5.2 Effectiveness of Selected Historical Tweets

To verify the effectiveness of the selection strategy used in our method, we applied the CROMA reinforcement learning module to other neural network-based baseline models, and then trained all the models on the benchmark to obtain new results. We compared the baseline models at two main settings. The first setting was training baseline models with the CROMA module on the benchmark with all 50 historical tweets of each user , which we denoted as "w/ RL." The second setting was training the same model without the CROMA module with all 50 historical tweets of each user, which we denoted as "w/o RL." We also trained baseline models on the benchmark with only five random selected historical tweets of each user, which we denoted as "w/o RL *." Table 3 shows a comparison

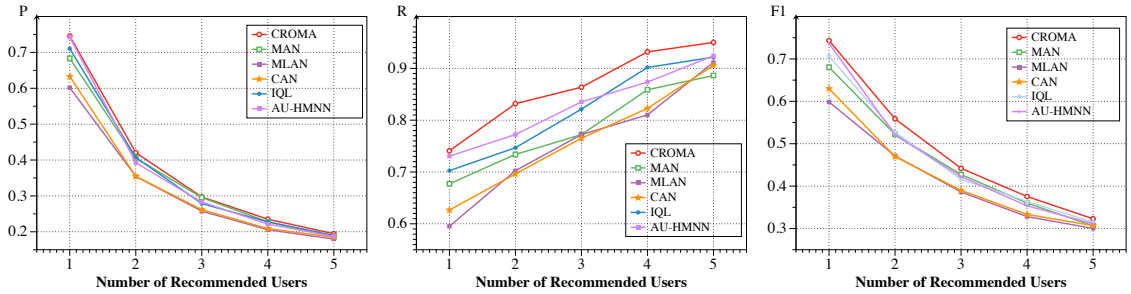

Figure 3: Precision, Recall and F-score with different number of recommended users.

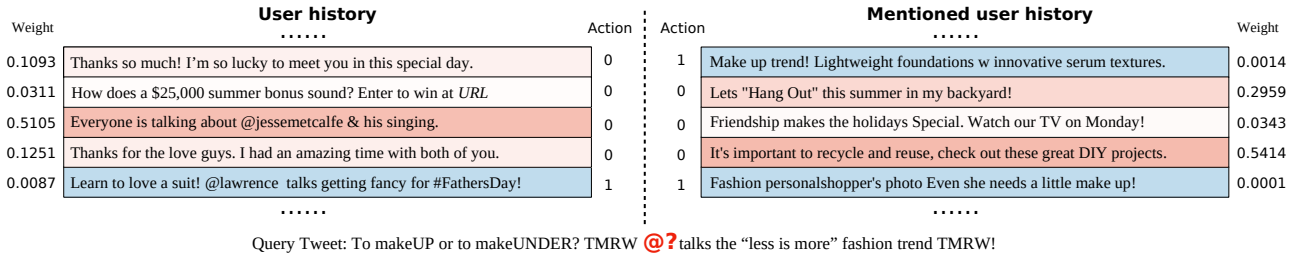

Figure 4: Tweet selection examples by different methods. For CROMA, 1 or 0 means the tweet is selected or not. For AU-HMNN, the value is the attention weight.

of the results, in which we can see that for hit@5, simply incorporating more historical tweets can not improve the performance steadily (as AU-HMNN), sometimes even result in performance degradation (as DAN). On the contrary, all the baseline methods perform better when they incorporate the CROMA module. The average improvement of baseline models with CROMA module for F1, Hits@3 and Hits@5 scores are roughly 0.91%, 1.73%, and 1.35%, respectively, which verifies that the selection strategy effectively promotes the performance of the mention recommendation task.

5.3 Indicator Tweets Discovery

Besides analyzing the effectiveness of our model, we also explored what kind of posts can indicate mention behaviors and why CROMA is better than an attention-based model. We trained the CROMA and the baseline models on the benchmark and then visualized the weights of the attention models and the behaviors of the CROMA. Figure 4 shows an example with respect to tweet selection. The tweet histories of the user and mentioned user contain several indicator tweets on the topic of makeup. However, because there are too many historical tweets, the attention model is easily negatively impacted by the presence of useless tweets. By contrast, the CROMA can correctly select relevant tweets and remove irrelevant tweets by an analysis of the current action and reverse action. Because it is less affected by the presence of irrelevant tweets, the model can make inferences with less confusion.

5.4 Ablation Test for State Designing

In this subsection, we evaluate the relative contribution of different factors of the state in CROMA to performance. The state in CROMA consists of the query-tweet representation t_q, current tweet representation h_i, global feature O_K, the representation of the indicator post set H_{indi}, and the representation of the irrelevant post set H_{irre}. Here, we test each factor by removing it and retraining the model on the benchmark from scratch, using the same random seed. The results are shown in Table 4.

Table 4: Ablation test on different state representations.

State	Precision	Recall	F1
ALL	74.55	74.09	74.32
$-H_{indi}, H_{irre}$	73.27	72.95	73.11
$-H_{indi}, H_{irre}, t_q$	71.13	70.76	70.94
$-O_K$	72.96	72.43	72.69
$-O_K + avg(H)$	71.59	70.98	71.28

Based on the results in the table, we can make three key points. **First**, the query-tweet representation is necessary. Because the model recommends mention candidates for the query tweet, if the CROMA does not use query tweet as a feature, it cannot take the correct action. Hence, the model without t_q performs 2.17% worse than the CROMA with respect to the F1 score. **Second**, the global feature O_K is important. At several initial steps, the CROMA has difficulty determining how to select the historical tweets based on the query tweet alone. The global feature O_K provides valuable information to the model, which increases the F1 score by 1.63%. However, if we simply average all the historical tweets, $avg(H)$, as a substitute for O_K, the results do not improve, which indicates that the feature O_K is more efficient than $avg(H)$. **Third**, historical actions can help to determine new actions. Reinforcement learning

is a process of sequential decision-making. The acquisition of historical actions made at an earlier time can help in formulating the current action. Hence, the model with H_{indi} and H_{irre} performs 1.21% better than that without.

6 CONCLUSIONS

In this work, we proposed a cooperative multi-agent approach for mention recommendation, which can effectively incorporate dozens of times as many historical tweets as previous approaches. To solve the problem of the same reward for all agents, we designed a reverse action mechanism to obtain differentiated rewards for two policy gradient agents to simultaneously select a small set of historical tweets from users and candidate mentioned users. We constructed and evaluated on a new mention recommendation dataset, in which each user contains dozens more historical tweets than previous works. Experimental results for the dataset show that the proposed method can significantly improve performance compared to those of other baseline approaches.

7 ACKNOWLEDGMENTS

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by China National Key R&D Program (No. 2018YFC0831105, 2017YFB1002104), National Natural Science Foundation of China (No. 61532011, 61751201), Shanghai Municipal Science and Technology Major Project (No.2018 SHZDZX01), STCSM (No.16JC1420401, 17JC1420200), ZJLab.

REFERENCES

[1] Kailong Chen, Tianqi Chen, Guoqing Zheng, Ou Jin, Enpeng Yao, and Yong Yu. 2012. Collaborative personalized tweet recommendation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 661–670.

[2] Zhuoye Ding, Xipeng Qiu, Qi Zhang, and Xuanjing Huang. 2013. Learning Topical Translation Model for Microblog Hashtag Suggestion.. In *IJCAI*. 2078–2084.

[3] Maxim Egorov. 2016. Multi-agent deep reinforcement learning.

[4] Jun Feng, Heng Li, Minlie Huang, Shichen Liu, Wenwu Ou, Zhirong Wang, and Xiaoyan Zhu. 2018. Learning to Collaborate: Multi-Scenario Ranking via Multi-Agent Reinforcement Learning. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1939–1948.

[5] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2017. Counterfactual multi-agent policy gradients. *arXiv preprint arXiv:1705.08926* (2017).

[6] Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Triantafyllos Afouras, Philip HS Torr, Pushmeet Kohli, and Shimon Whiteson. 2017. Stabilising Experience Replay for Deep Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. 1146–1155.

[7] Yuyun Gong and Qi Zhang. 2016. Hashtag Recommendation Using Attention-Based Convolutional Neural Network.. In *IJCAI*. 2782–2788.

[8] Yeyun Gong, Qi Zhang, Xuyang Sun, and Xuanjing Huang. 2015. Who Will You@?. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 533–542.

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[10] Haoran Huang, Qi Zhang, Xuanjing Huang, et al. 2017. Mention recommendation for Twitter with end-to-end memory network. In *Proc. IJCAI*, Vol. 17. 1872–1878.

[11] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[12] Vijay R Konda and John N Tsitsiklis. 2000. Actor-critic algorithms. In *Advances in neural information processing systems*. 1008–1014.

[13] Ioannis Konstas, Vassilios Stathopoulos, and Joemon M Jose. 2009. On social networks and collaborative recommendation. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 195–202.

[14] Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl. 2009. Latent dirichlet allocation for tag recommendation. In *Proceedings of the third ACM conference on Recommender systems*. ACM, 61–68.

[15] Quanle Li, Dandan Song, Lejian Liao, and Li Liu. 2015. Personalized mention probabilistic ranking–recommendation on mention behavior of heterogeneous social network. In *International Conference on Web-Age Information Management*. Springer, 41–52.

[16] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. In *NIPS*. 289–297.

[17] Hao Ma, Irwin King, and Michael R Lyu. 2009. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 203–210.

[18] Renfeng Ma, Qi Zhang, Jiawen Wang, Lizhen Cui, and Xuanjing Huang. 2018. Mention Recommendation for Multimodal Microblog with Cross-attention Memory Network. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, New York, NY, USA, 195–204. https://doi.org/10.1145/3209978.3210026

[19] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[20] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[21] Seungwhan Moon, Leonardo Neves, and Vitor Carvalho. 2018. Multimodal named entity recognition for short social media posts. *arXiv preprint arXiv:1802.07862* (2018).

[22] Mor Naaman, Jeffrey Boase, and Chih-Hui Lai. 2010. Is it really about me?: message content in social awareness streams. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*. ACM, 189–192.

[23] Hyeonseob Nam, Jung-Woo Ha, and Jeonghee Kim. 2017. Dual Attention Networks for Multimodal Reasoning and Matching. In *CVPR*. 299–307.

[24] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[25] Soumajit Pramanik, Mohit Sharma, Maximilien Danisch, Qinna Wang, Jean-Loup Guillaume, and Bivas Mitra. 2018. Easy-Mention: a model-driven mention recommendation heuristic to boost your tweet popularity. *International Journal of Data Science and Analytics* (2018), 1–17.

[26] Richard S Sutton and Andrew G Barto. 1998. *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge.

[27] Yuri Takhteyev, Anatoliy Gruzd, and Barry Wellman. 2012. Geography of Twitter networks. *Social networks* 34, 1 (2012), 73–81.

[28] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. 2017. Multiagent cooperation and competition with deep reinforcement learning. *PloS one* 12, 4 (2017), e0172395.

[29] Liyang Tang, Zhiwei Ni, Hui Xiong, and Hengshu Zhu. 2015. Locating targets through mention in Twitter. *World Wide Web* 18, 4 (2015), 1019–1049.

[30] Kagan Tumer and Adrian Agogino. 2007. Distributed agent-based air traffic flow management. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. ACM, 255.

[31] Beidou Wang, Can Wang, Jiajun Bu, Chun Chen, Wei Vivian Zhang, Deng Cai, and Xiaofei He. 2013. Whom to mention: expand the diffusion of tweets by@ recommendation on micro-blogging systems. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1331–1340.

[32] Kai Wang, Weiyi Meng, Jiang Bian, Shijun Li, and Sha Yang. 2018. Spatial context-aware user mention behavior modeling for mentionee recommendation. *Neural Networks* 106 (2018), 152–167.

[33] Kai Wang, Weiyi Meng, Shijun Li, and Sha Yang. 2019. Multi-Modal Mention Topic Model for mentionee recommendation. *Neurocomputing* 325 (2019), 190–199.

[34] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3-4 (1992), 279–292.

[35] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.

[36] Serena Yeung, Vignesh Ramanathan, Olga Russakovsky, Liyue Shen, Greg Mori, and Li Fei-Fei. 2017. Learning to Learn from Noisy Web Videos. In *2017 CVPR*. IEEE, 7455–7463.

[37] Dongfei Yu, Jianlong Fu, Tao Mei, and Yong Rui. 2017. Multi-level attention networks for visual question answering. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 4187–4195.

[38] Wei Zhang, Jianyong Wang, and Wei Feng. 2013. Combining latent factor model with location features for event-based group recommendation. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 910–918.

[39] Ge Zhou, Lu Yu, Chu Xu Zhang, Chuang Liu, Zi Ke Zhang, and Jianlin Zhang. 2016. A Novel Approach for Generating Personalized Mention List on Micro-Blogging System. In *IEEE International Conference on Data Mining Workshop*.