Noname manuscript No.
(will be inserted by the editor)

RLINK: Deep Reinforcement Learning for User Identity Linkage

Xiaoxue Li · Yanan Cao✉ · Qian Li · Yanmin Shang✉ · Yangxi Li · Yanbing Liu · Guandong Xu

Received: date / Accepted: date

Abstract User identity linkage is a task of recognizing the identities of the same user across different social networks (SN). Previous works tackle this problem via estimating the pairwise similarity between identities from different SN, predicting the label of identity pairs or selecting the most relevant identity pair based on the similarity scores. However, most of these methods fail to utilize the results of previously matched identities, which could contribute to the subsequent linkages in following matching steps. To address this problem, we transform user identity linkage into a sequence decision problem and propose a reinforcement learning model to optimize the linkage strategy

Xiaoxue Li
College of Cyberspace Security, University of Chinese Academy of Sciences, Beijing, China
Institute of Information EngineeringChinese Academy of Sciences, Beijing, China
E-mail: lixiaoxue@iie.ac.cn

✉Yanan Cao
Institute of Information EngineeringChinese Academy of Sciences, Beijing, China
College of Cyberspace Security, University of Chinese Academy of Sciences, Beijing, China
E-mail: caoyanan@iie.ac.cn

Qian Li
University of Technology Sydney, Australia, Sydney
E-mail: Qian.Li@uts.edu.au

✉Yanmin Shang
Institute of Information EngineeringChinese Academy of Sciences, Beijing, China
E-mail: shangyanmin@iie.ac.cn

Yangxi Li
National Computer network Emergency Response technical Team, Beijing, China
E-mail: liyangxi@outlook.com

Yanbing Liu
Institute of Information EngineeringChinese Academy of Sciences, Beijing, China
E-mail: liuyanbing@iie.ac.cn

Guandong Xu
University of Technology Sydney, Australia, Sydney
E-mail: Guandong.Xu@uts.edu.au

from the global perspective. Our method makes full use of both the social network structure and the history matched identities, meanwhile explores the long-term influence of processing matching on subsequent decisions. We conduct extensive experiments on real-world datasets, the results show that our method outperforms the state-of-the-art methods.

Keywords Social network, Reinforcement Learning, User Identity Linkage, Markov Decision Process

1 Introduction

User Identity Linkage (UIL), which aims to recognize the identities (accounts) of the same user across different social platforms, is a challenging task in social network analysis. Nowadays, many users participate in multiple on-line social networks to enjoy more services. For example, a user may use Twitter and Facebook at the same time. However, on different social network platforms, the same user may register different accounts, have different social links and deliver different comments. If different social networks could be integrated together, we could create an integrated profile for each user and achieve better performance in many practical applications, such as link prediction and cross-domain recommendation. So, UIL has recently received increasing attention both in academia and industry.

Most of previous works consider UIL as an one-to-one alignment problem [5,11,33,35], i.e., each user has at most one identity in each social network. Matching models [25,24], label propagation algorithms [4,8,14,19,23, 38] and ranking algorithms [11,31,33,37] are commonly used to address this task. These methods generally calculate pairwise similarity between identities and select the most relevant identity pairs according to the similarity score. In more recent works, identity similarity is computed based on user embedding [33,11,37,41], which encodes the main structure of social networks or other features into a low-dimensional and density vector. In most of these methods, each identity pair is matched independently, i.e., one predicted linkage of the identity pair would not be effected by any other.

However, the predicted linkages are inter-dependent and thus have a long-term influence on the subsequent one. This long-term influence is two-fold: (i) if the preceding user identity linking is right, they would bring in positive auxiliary information to guide the following linkage. For example, if two user from two different social networks share the same friends (their friends have been matched), they are more likely to be matched in the subsequent linkage, as shown in Figure 1; (ii) the previously matched user identity could not be chosen in the subsequent matching process according to the *one-to-one constraint* [5], Although some previous label propagation based works have made primary attempts on using this influence, they just gave the fixed greedy strategy to iteratively select the candidate identity pair [14,19,23,38]. Neither of them re-calculated the pairwise similarity or dynamically adjusted the linkage strategy after each matching step due to the high complexity.

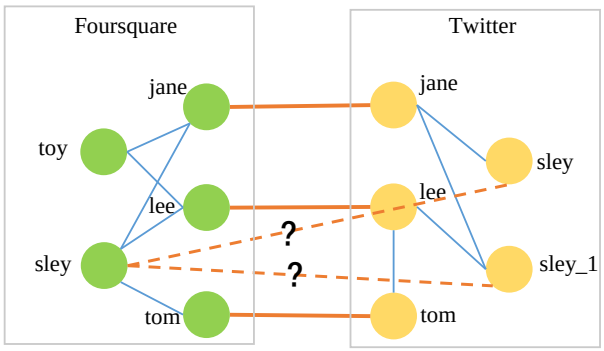


Fig. 1 An example of User Identity Linkage. The blue link represents friend relation in social network, and orange line represents matched identity pair. User identity sley (in Foursquare) has two candidate identities in Twitter: sley and sley_1. If considering previously matched information, sley in Foursquare is similar to sley_1 in Twitter because they share more similarity friends.

In order to model this long-term influence effectively, we novelly consider UIL as a Markov Decision Process and propose a deep reinforcement learning framework (RLink) to automatically match identities in two different social networks. Figure 2 illustrates the overall RLink process. One state consists of three components, i.e. two social network structures and previously matched identity pairs. According to the current state, the agent performs an **action**. After the action is performed, the **state** would be changed at the next time and a **reward** would be fed to the agent to adjust its policy. Because the action space is large and dynamic in the UIL process, we adapt an Actor-Critic framework [29], in which the actor network generates a deterministic action based on current state and the critic network evaluates the quality of this action-state pair. Concretely, for each state, the actor firstly encodes the network structure and history decisions to a latent vector representation, and then decodes this vector into an action in the identity embedding space. Based on this, our model could generate an identity pair from the candidates automatically.

By directly optimizing the overall evaluation metrics, deep RL model performs much better than models with loss functions that just evaluate a particular single decision [9,35]. However, there are spots of attempts on applying RL in the social network analysis field [7,28,22]. To the best of our knowledge, we are the first to design a deep RL model for user identity linkage. Our RL model is able to produce more accurate results by fully exploring the long-term influence of independent decisions.

The contributions of this paper can be summarized as follows:

– We are the first to consider UIL as a sequence decision problem and innovatively propose a deep reinforcement learning based model for this task, which generates the matching sequence automatically generate.

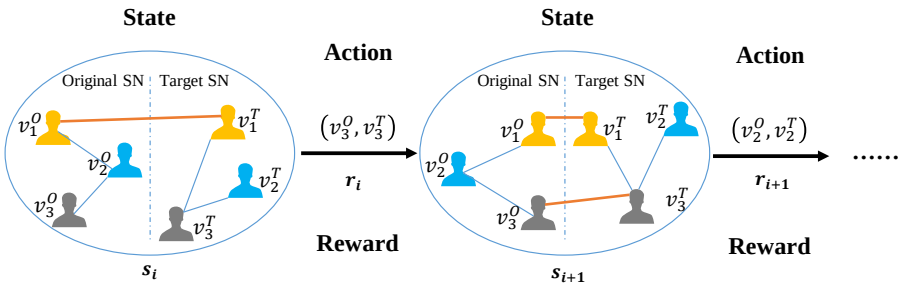


Fig. 2 The Procedure of the proposed RLink of Reinforcement Learning based User Identity Linkage. The blue link represents friend relation in social network, and orange line at state S_i represents matched identity pair. At time i, agent generates a pair of matching identities as an action according to current state. After agent performs this action, the state would be changed at time $i+1$ and next action can be generated based on S_{i+1}.

- The proposed model makes full use of the previous matched identity pairs which may have the long-term influence on the subsequent linkage. This allows to link user identity from a global perspective.
- Extensive experiments are conducted on three pairs of real-word datasets to show that our method achieves better performance than the state-of-the-art solutions for user identity linkage.

2 The Proposed Model

2.1 Preliminary

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents an online social network, where $\mathcal{V} = \{v_1, v_2, ..., v_N\}$ is the set of user identities and $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ is the set of links in the network. Given two online social network $\mathcal{G}^O$ (original network) and $\mathcal{G}^T$ (target network), the task of user identity linkage is to identify hidden user identities pairs across $\mathcal{G}^O$ and $\mathcal{G}^T$. Here, we have a set of node pair (v_i^O, v_j^T) between $\mathcal{G}^O$ and $\mathcal{G}^T$ to represent given alignment information (ground-truth), denoted as $\mathcal{B}$

Table 1 Statistics of experimental datasets

$\mathcal{G}^O$, $\mathcal{G}^T$	the input original/target network
$\mathcal{V}^O$, $\mathcal{E}^O$	the node/edge set of $\mathcal{G}^O$
$\mathcal{V}^T$, $\mathcal{E}^T$	the node/edge set of $\mathcal{G}^T$
$\mathcal{B}$	the alignment information of $\mathcal{G}^O$ and $\mathcal{G}^T$
$\mathcal{S}$,$\mathcal{A}$	the set of state/action
$\mathbf{u}_k$	the initial embedding of node v_k
e^O, e^T	the embedding of origin/target network embedding
s^{net}, s^{pair}	the representation of current network structure/history matched identity pairs
$\mathbf{g}_i, x_i$	the feedback/representation of i-th action (one history matched pair)

In this work, we consider UIL task as a markov decision process in which the linkage agent interacts with environment over a sequence of steps. We use

$s \in \mathcal{S}$ to denote the current state, which consists of the network structure and matched identity pairs. Action denoted as $a \in \mathcal{A}$ is a pair of identities. The action space of UIL is comprised of all potential identity pairs, the size of which is $|\mathcal{V}^O| \times |\mathcal{V}^T|$. At each step, the agent generates an action a based on s, and would receive a reward $r(s, a)$ according to the given alignment information. The goal of RLink is to find a linkage policy $\pi : \mathcal{S} \leftarrow \mathcal{A}$, which can maximize the cumulative reward for linkage.

Due to the high dimension and dynamic property og action space, the policy-based RL [3,6] which computes probability distribution of every action and the Q-learning [16,30] which evaluates the value of each potential action-state pair are time-consuming. To reduce the high computational cost, we adapt the Actor-Critic framework. The actor inputs the current state s and aims to output a deterministic action, while the critic inputs only this state-action pair rather than all potential state-action pairs. The architecture of the Actor-Critic network is shown in Figure 3.

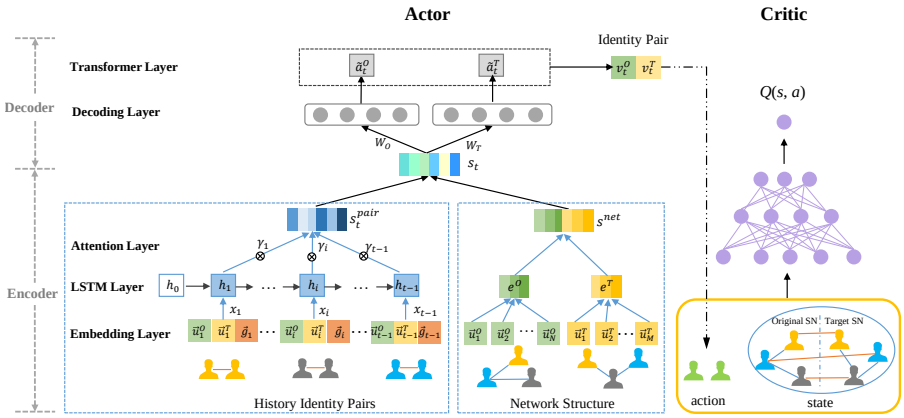


Fig. 3 The framework of Actor-Critic network, where the actor is comprised of an Encoder-Decoder architecture and the critic is DQN. The inputs of Actor are history identity pairs and network structure, where history identity pairs were generated by our Actor-Critic network before current epoch (See Eq.(4)). Remarkably, h_0 is a zero vector. Then this action and current state are input onto the Critic to evaluate the quality of this action.

2.2 Architecture of Actor Network

The goal of our Actor network is to generate an action (one matching identity pair) according to the current state (network structure and history identity pairs). We propose an Encoder-Decoder architecture to achieve this goal.

2.2.1 Encoder for Current State

Encoder aims to generate the representation of current state, which contains two types of information: network structure and historical identity pairs. In order to integrate these information, we apply two encoding mechanisms to respectively encode the network structure and the matched pairs, as shown in Figure 3.

In our model, each identity v_k ($k \in (1, N)$) is represented as a low dimensional and dense vector, which is denoted as $\mathbf{u}_k$ and $\mathbf{u}_k \in R^d$ (d is the dimension of the identity embedding). These identity embeddings are pre-trained by Node2vec [1]. To represent the social network structure, we weighted sum all identity embedding to generate the network embedding e, which is inspired by [12].

$$e = \sum_{v_k \in \mathcal{V}} \alpha_k \mathbf{u}_k, \tag{1}$$

where α_k denotes the weight of the identity v_k. Here, we define α_k as $\frac{\zeta}{\zeta + d(v_k)}$ (similar to [12]), where ζ is a constant and $d(v_k)$ represents the degree of identity v_k. We denote the representation of original network and target network as e^O and e^T, then we concatenate them to produce s^{net} to represent network structure information, where $s^{net} \in R^{2d}$. That is:

$$s^{net} = concat(e^O, e^T).$$

At time t, we need to encode all previously matched identity pairs from time 1 to $t-1$ and produce an action sequence $\{a_1, a_2, ..., a_{t-1}\}$. In each action a_i ($i \in [1, t-1]$), the identity from $\mathcal{G}^O$, $\mathcal{G}^T$ are respectively denoted as v_i^O and v_i^T, and their embedding representations are denoted as $\mathbf{u}_i^O$ and $\mathbf{u}_i^T$. Besides, we take the feedback of a_i, which is denoted as g_i, into account. g_i is a one-hot vector, in which the value of each dimension is equal to the immediate reward at the corresponding time i. Through a neural network layer, we get the encoding vector of g_i as $\mathbf{g}_i$:

$$\mathbf{g}_i = \varphi(W_G g_i + b_G), \tag{2}$$

where $g_i \in R^{|g|}$ and $\mathbf{g}_i \in R^{|G|}$, and $R^{|g|}$ is equal to the number of steps in each episode and $R^{|G|}$ depends on the dimension of W_G.

Then, we get the representation of one history matched pair at time i by concatenating $\mathbf{u}_i^O, \mathbf{u}_i^T$ and $\mathbf{g}_i$ which is denoted as x_i:

$$x_i = concat(\mathbf{u}_i^O, \mathbf{u}_i^T, \mathbf{g}_i). \tag{3}$$

where the dimensional of x_i is $|G| + 2d$.

In order to capture the long-term influence of the previously matched pairs, we use a Long Short-Term Memory (LSTM) network to encode the history linkage sequence into a fixed-size vector:

$$h_i = LSTM(x_i, h_{i-1}), \tag{4}$$

Furthermore, to distinguish different contributions of the previous actions, we employ attention mechanism [2], which allows model to adaptively focus on different parts of the input:

$$s_t^{pair} = \sum_{i=1}^{t-1} \gamma_i h_i, \tag{5}$$

where the dimension of s_t^{pair} is equal to s^{net}, and we leverage an attention mechanism[15] to compute γ_i from the hidden state h_i

$$\gamma_i = \frac{exp(w_\gamma h_i + b_\gamma)}{\sum_j exp(w_\gamma h_j + b_\gamma)}$$

Then, the embedding of current state can be represented as follows:

$$s_t = s_t^{pair} + s^{net}, \tag{6}$$

where $s_t \in R^{2d}$, s_t^{pair} and s^{net} represent the embedding of history matching information and network structure respectively.

It is noteworthy that, at time 1, the current state just contains two social network structures because there is not matched identity pair. That is to say, $h_0 = 0$, s_t^{pair} is equal to zero vector and s_t is equal to s^{net}.

Note that, the decoded $\widetilde{a}_t^O$ and $\widetilde{a}_t^T$ maybe not in the identity embedding space. Thus we need to map them into the real embedding space via the transformation [36]. As mentioned above, the action space is very large. In order to correctly map $\widetilde{a}_t$ into the validate identity, we select the most similar $\mathbf{u}_t \in U$ as the valid identity-embedding. In this work, we compute the cosine similarity to get the valid identity in given networks:

$$v_t^O = \max_{v^O \in \mathcal{V}^O} \left(\mathbf{u}_t^O\right)^{\mathrm{T}} \cdot \frac{U^O}{\|U^O\|}, \tag{7}$$

$$v_t^T = \max_{v^T \in \mathcal{V}^T} \left(\mathbf{u}_t^T\right)^{\mathrm{T}} \cdot \frac{U^T}{\|U^T\|}, \tag{8}$$

where $\{v_t^O, v_t^T\}$ represents the valid identity pair a_t. We pre-compute the value of $\frac{U^O}{\|U^O\|}$ and $\frac{U^T}{\|U^T\|}$ to decrease the computational cost. Since the alignment identities could not be chosen in the following steps, we ignore those identities if they are correctly matched in the previous steps.

2.2.2 Reward

The agent generates the new identity pair and receives the immediate reward r_{tm} from networks, which is also the feedback of this new identity pair,

$$r_{tm} = \begin{cases} 1, & a_i \in Groundtruth; \\ -1, & \text{else}, \end{cases}$$

where *Groundtruth* is a set of known aligned identity pairs. Since current action result has a long-term impact on subsequent decisions, we introduce a discount factor λ to metric the weight of reward:

$$r_t = \lambda_t r_{tm}, \tag{9}$$

where $\lambda_t \in \{0, 1\}$ represents how much influence the action a_t will generate on the following steps. Simply, λ_t is defined as $\frac{1}{t}$, and t represents the current time stamp.

2.3 Architecture of Critic Network

Critic Network aims to judge whether the action a_t generated by Actor fits the current state s_t. Generally, the Critic is designed to learn a Q-value function $Q(s, a)$, while the actor updates its' parameters in a direction of improving performance to generate next action according to $Q(s, a)$ in the following steps. However, in real UIL, the state and action space is enormous and many state-action pairs may not appear in the real traces, which makes it hard to update their Q-values. Thus, we choose Deep Neural Network as an approximator to estimate the action-value function. In this work, we refer to a neural network as Deep Q-value function (DQN)[17].

Firstly, we need to feed user's current state s and action a into the DQN. To generate user's current state s, the agent follows the same strategy from Eq.(1) to Eq.(6). As for action a, we utilize the same strategy in decoder to compute a low-dimensional dense action vector a. Then, this action is evaluate by the DQN, which returns the Q-value of this state-action pair $Q(s, a)$.

2.4 Training and Test

Generally, we utilize DDPG[13] algorithm to train the proposed Actor-Critic framework, which has four neural networks: critic, actor, critic-target and actor-target, where critic-target and actor-target networks are the copy of critic and actor respectively. The Critic is trained by minimizing the mean squared error loss with the corresponding target given by:

$$L(\beta_c) = E_{s,a,R,s'}[(R + \rho Q_{\beta'_c}(s', f_{\theta_{\pi'}}(s')) - Q_{\beta_c}(s, a))^2], \tag{10}$$

where ρ is the discount factor in RL, R is the accumulative reward and s' is the previous state, $f_{\theta_{\pi'}}(s') = a'$. Besides, β_c and θ_π represent all parameters in Critic and Actor respectively, and f_{θ_π} represents the policy. The Critic is trained from samples stored in a replay buffer[17]. Similarity, actions also stored in the replay buffer generated by the strategy in Actor decoder section. This allows the learning algorithm to dynamic leverage the information of which action was actually executed to train the critic.

The first term $R + \rho Q_{\beta'_c}(s', f_{\theta_{\pi'}}(s'))$ in Eq.(9) is the output of target, namely y, for current iteration. And parameters from the previous iteration

$\theta_{\pi'}$ are fixed when optimizing the loss function $L(\beta_c)$. Computing the full expectations' gradient are not efficient. Thus, we optimize the loss function by Stochastic Gradient Descent (SGD). The derivatives of loss function $L(\beta_c)$ with respective to parameters β_c are presented as follows:

$$\nabla_{\beta_c} L(\beta_c) = E_{s,a,r,s'}[R + \rho Q_{\beta_c}(s', f_{\theta_{\pi'}}(s')) - Q_{\beta_c}(s,a) \nabla_{\beta_c} Q_{\beta_c}(s,a)] \quad (11)$$

The Actor is updated by using the policy gradient:

$$\nabla_{\theta_\pi} \approx E_s[\nabla_a Q_{\beta_c}(s,a) \nabla_{\theta_\pi} f_{\theta_\pi}] \quad (12)$$

The training algorithm for the proposed framework RLink is presented in Algorithm 1. In each iteration, there are two stages, i.e., 1) generating an action (lines 8-11), and 2) parameter updating (lines 13-17). For generating an action, given the current state s_t, the agent firstly encode current state as a vector (line 8) and then generate a pair of nodes according to this vector (line 9); then the agent observes the reward r_t and update state to s_{t+1} (line 10); finally the agent stores transitions $(s_t, a_t, r_t, s_(t+1))$ into replay buffer $\mathcal{D}$. For the parameter updating stage: the agent samples mini-batch of transitions (s, a, r, s') from $\mathcal{D}$ (line 13), and then updates parameters of Actor and Critic following a standard DDPG procedure (lines 14-16). Finally, the parameters of target network $\beta_{c'}$ and $\theta_{\pi'}$ are updated via the soft update way (line 17).

To evaluate the performance of our model, the test procedure is designed as an online test method, which is similar to the action generation stage in the training procedure. After the training procedure, proposed framework RLink learns parameters θ_π and β_c. In each iteration, the agent generates a pair of identity a_t following the trained policy f_{θ_π}. And then the agent receives the reward r_t from networks and updates the state to s_{t+1}.

Algorithm 1 The RLink algorithm.

1: Initial Actor network f_{θ_π} and critic network Q_{β_c} with random weights;
2: Initial target network $f_{\theta_{\pi'}}$ and $Q_{\beta_{c'}}$ with weights $\theta_{\pi'} \leftarrow \theta_\pi$, $\beta_{c'} \leftarrow \beta_c$
3: Initial the capacity of replay buffer $\mathcal{D}$
4: **for** $session = 1, \Gamma$ **do**
5: Receive initial observation state s_1
6: **for** $t = 1, T$ **do**
7: **Stage 1: Generating an action**
8: Encode current state s_t according to Eq.(1) to Eq.(6)
9: Generate the valid identity pair according Eq.(7) and Eq.(8) as action a_t
10: Observe the reward r_t according to Eq.(9) and new state s_{t+1}
11: Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
12: **Stage 2: Parameter updating**
13: Sample mini-batch $\mathcal{N}$ transitions (s, a, r, s') in $\mathcal{D}$
14: Set $R + \rho Q_{\beta_c}(s', f_{\theta_{\pi'}}(s'))$
15: Update Critic by minimizing $\frac{1}{N}\sum_n (y - Q_{\beta_c}(s,a))^2$ according to Eq.(11)
16: Update Actor using the sampled policy gradient according to Eq.(12)
17: Update the target network: $\theta_{\pi'} \leftarrow \tau\theta_\pi + (1-\tau)\theta_{\pi'}$; $\beta_{c'} \leftarrow \tau\beta_c + (1-\tau)\beta_{c'}$
18: **end for**
19: **end for**

Table 2 Statistics of experimental datasets

Dataset	User Identities	Social links	Ground truth
Foursquare	5,120	76,972	1,609
Twitter	5,313	164,920	
Last.fm	136,420	1,685,524	1,381
Myspace	854,498	6,489,736	
Aminer	1,053,188	3,916,907	4,153
Linkedin	2,985,414	25,965,384	

3 Experiment

In this section, we compare our RLink with the state-of-the-art methods on user identity linkage task.

3.1 Experiment Setup

3.1.1 Datasets

In order to verify our method in different types of networks, we conduct experiments on the following datasets: Foursquare-Twitter, Last.fm-Myspace and Aminer-Linkedin, which are benchmark datasets in the UIL task. The first dataset is provided by [34] and other datasets are collected by [35]. These datasets are introduced as follows and the statistic information is shown in Table 2 which considers social links as user identities' feature.

- Foursquare-Twitter is a pair of social networks, where users share their current location and other information with others.
- Last.fm-Myspace is a pair of online social networks, where users could search music and share their interested music with others.
- Aminer-Linkedin is a pair of citation networks which contains users' academic achievements and users could search interested community.

3.1.2 Comparative Methods

To evaluate the performance of RLink for user identity linkage, we choose the following state-of-the-art methods for comparison, including:

- IONE [11] IONE predicts anchor links by learning the followership embedding and followeeship embedding of a user simultaneously.
- DeepLink [37] DeepLink employs unbiased random walk to generate embeddings, and then use MLP to map users.
- MAG [31] MAG uses manifold alignment on graph to map users across networks.

- PAAE [41] PAAE employs an adversarial regularization to capture the robust embedding vectors and maps anchor users with an alignment auto-encoders.
- SiGMa [9] SiGMa is designed to align two given networks by propagating the confidence score to the matching network. We use the name matching method to generate the seed set and utilize the output scores of SVM as the pairwise similarity. Note that SiGMa is an unsupervised method.
- SDM: This method is a simpler deep model, which is similar to typical sequence prediction methods [10] and utilizes LSTM to process sequence matching. The matching sequence is generated via ranking the similarity of embedding. At each step i, LSTM predicts an user identity in Target SN based on current user identity embedding and previous hidden information (h_{i-1}). Finally, the L_2 loss would guide the training and testing process of this model.

3.1.3 Evaluation Metrics

To perform the user identity linkage, we utilize three standard metrics [26] to evaluate the performance, including *Precision@k (P@k)*, recall and *MAP*. Note that higher the value for these metrics indicates the better performance.

Precision@k evaluates the linking accuracy, and is defined as:

$$P@k = \sum_{i}^{n} \mathbf{1}_i\{success@k\}/n, \tag{13}$$

where $\mathbf{1}_i\{success@k\}$ measures whether the positive matching identity exists in $top-k(k <= n)$ list, and n is the number of testing anchor nodes.

The recall is the fraction of the number of real corresponding user pairs that have been found over the total amount of real matched user pairs (Ground-truth $\mathcal{B}$).

$$Recall = \frac{success\ matched\ pairs}{Real\ user\ pairs\ in\ \mathcal{B}} \tag{14}$$

MAP is used for evaluating the ranking performance of the algorithms, defined as:

$$MAP = (\sum^{n} \frac{1}{ra})/n, \tag{15}$$

where ra is the rank of the positive matching identity and n is the number of testing anchor nodes.

3.1.4 Hyper-parameter setting

For each pair of networks, we first resort the ground truth data set by identity number and then use the first r as the training data and the remaining $1-r$ as the testing data, where r means the training ratio. The dimension of the identity embedding is set to 128, and the sizes of inputs for encoder and decoder

Table 3 Performance comparison on user identity linkage

Dataset	Metrics	MAG	IONE	SiGMa	SDM	DeepLink	PAAE	RLink
Foursquare Twitter	*P@1*	6.38	22.38	-	51.92	34.47	21.98	**75.93**
	P@9	17.05	46.38	-	51.92	66.09	47.62	**75.93**
	MAP	-	32.79	-	51.92	47.78	40.68	**75.93**
Last.fm Myspace	*P@1*	-	29.57	**95.65**	72.72	-	28.76	85.71
	P@9	-	53.21	**95.65**	72.72	-	52.97	85.71
	MAP	-	47.61	**95.65**	72.72	-	46.25	85.71
	recall	-	-	32.86	6517	-	-	**77.68**
Aminer Linkedin	*P@1*	-	31.76	88.50	82.50	-	30.96	**92.85**
	P@9	-	59.28	88.50	82.50	-	59.42	**92.85**
	MAP	-	50.12	88.50	82.50	-	52.76	**92.85**
	recall	-	-	47.39	79.46	-	-	**91.54**

are the same, i.e., $|U| = |G| = 128$. And the weighting parameter ζ is fixed to 10^{-3}. For parameters in Actor, the number of LSTM cell units is set to 256 and batch size is 64. In Critic, the number of hidden layer is 4. The batch size in replay buffer is 64 and we set 200 sessions in each episode. Besides, in training procedure, we set learning rate $\eta = 0.001$, discount factor $\rho = 0.9$, and the rate of target networks soft update $\tau = 0.001$. When evaluating the sensitively of one of hyper-parameters, other hyper-parameter are set to the optimal one.

3.2 Comparisons and Analysis

We compare our proposed RLink model with the following recent user identity linkage methods. Note that SiGMa, SDM and RLink are prediction methods which determine whether two user identities from original and target are matched or not. Therefore, the evaluation of those methods becomes $P@k = P@1 = MAP$. From Table 3, we can see that:

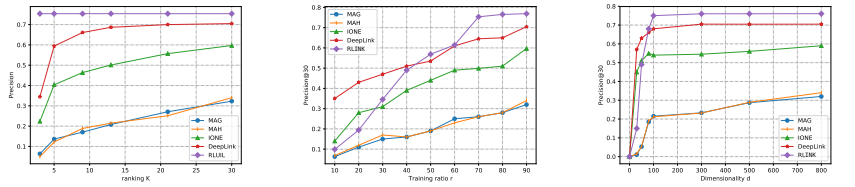
- RLink is significantly better than previous user identity linkage methods. This demonstrates that our method which considers UIL as a sequence decision problem and makes decisions from a global perspective is useful. Besides, reinforcement learning based methods, which learns to directly optimize the overall evaluation metrics, performs better than other deep learning methods, such as DeepLink and PAAE.
- SDM and SiGMa achieve higher precision than other baselines, which demonstrates that considering the influence of previous matched information is beneficial for UIL task. Although SigMa achieves the highest precision on Last.fm-Myspace, it suffers from a low recall due to its fixed greedy matching strategy. By contrast, SDM and RLink promote approximately 30% and 40% recall over SigMa on respectively, which proves previous matched result has a long-time impact on the subsequent matching process.
- IONE, PAAE and DeepLink perform better than MAH and MAG, which means that considering more structural information, such as followership/

followee-ship and global network structure, could achieve higher precision. Besides, the precision of PAAE and DeepLink are higher than IONE, which demonstrates that considering global network structure and utilizing deep learning model could improve the performance of UIL.
– Besides, we note that all methods perform better on Aminer-Linkedin than Last.fm-Myspace, maybe because the scale of the training data in Aminer-Linkedin is larger.

3.3 Discussion

3.3.1 Parameter Sensitivity

In this section, we analyze the sensitivity of three parameters which are different K in $P@K$, training ratio r and embedding dimension d. The performance for those parameters on all three testing datasets is similar. We present the performance on Twitter-Foursquare due to the limited spaces.



(a) different K (b) Training Ratio (c) Embedding Dimension

Fig. 4 Detailed Performance Comparison on Twitter-Foursquare Dataset

Precision on different K. For different K in $P@K$, we report the precision of different methods on variable K between 1 and 30, as shown in Figure 4a. RLink outperforms all the comparison methods consistently and significantly given different @K settings. IONE performs better than MAG and MAH, showing that constructing the incidence matrices of the hypergraph could fail to differentiate the follower-ship and followee-ship. Both RLink and DeepLink perform better than MAG, MAH and IONE, showing that deep learning methods could extract more node features for UIL than traditional methods. The precision of most of the testing methods increases significantly with the rise of K until $K = 20$, indicating that most of ranking methods can match identity in $Top - 20$.

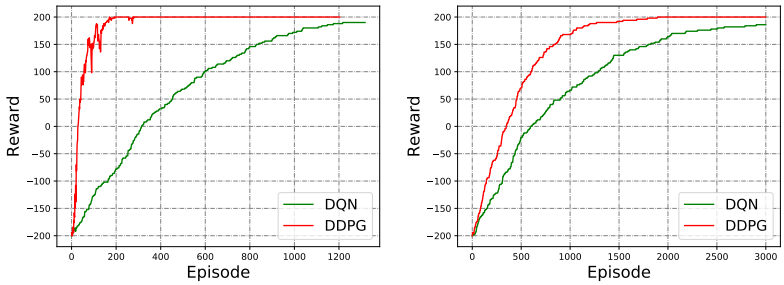
Precision on different training ratio r. For different training ratios, we report the Precision@30 of different methods on variable training ratios between 10% and 90%. RLink outperforms all the comparison methods when the ratio increase to 60%. The ratio of anchor nodes used for training greatly affects the performance of RLink. Especially for ratio settings as 60% to 70%, the performance enhancement is significant. While the results show that with

the increase of training data, the precision of UIL firstly increases significantly and then does not increase drastically as the ratio increases to 70%. And the comparative methods achieve good performance when the training ratio is around 90%, which demonstrates the robust adaptation of our RLink. Besides, from Figure 4b, we found that DeepLink performs better than RLink when the training ratio is less then 40%, indicating that RLink might need more train dataset than DeepLink.

Precision on different embedding dimension d**.** According to Figure 4c, both MAG and MAH achieve good performance when the dimensionality is around 800, while RLink, IONE and DeepLink achieve good performance when the dimensionality is around 100. The complexity of the learning algorithm highly depends on the dimensionality of the subspace. And low dimensional representation also leads to an efficient relevance computation [11]. Therefore, we conclude that RLink, IONE and DeepLink are significantly more effective and efficient than MAH and MAG. Besides, RLink achieves the best results when the dimensionality is bigger than 80.

3.3.2 Evaluation of Actor-Critic Framework

In this paper, we use DDPG algorithm to train the RLink model. To evaluate the effectiveness of the Actor-Critic framework, we compare the performance of DDPG with DQN. From Figure 5, we can see that DQN performs similar to DDPG, but the training speed of DQN is much slower. As shown in Figure 5a, DQN needs 1200 iterations to converge, which is almost four times of the iterations DDPG needs. In Linked-Aminer, as shown in Figure 5b, both DDPG and DQN need more training iterations, i.e., 750 and 2000 episodes respectively, when the size of action space increase to 4153×4153. Besides, DQN performs worse than it does in Last.fm-Myspace, which demonstrates that DQN is not suitable for the large action space. In summary, DDPG achieves more accuracy and faster training speed than DQN, which indicates that Actor-Framework is suitable for UIL with enormous action space.

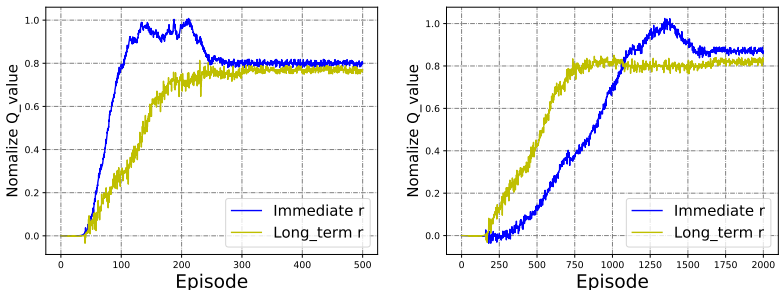


(a) Last.fm-Myspace (b) Linkedin-Aminer

Fig. 5 Comparison between DDPG and DQN: The x-axis shows the training episodes. The y-axis shows the total reward of each episode. Red line represents DDPG and green Line is DQN.

3.3.3 Effect of Long-term Reward

To evaluate the effectiveness of long-term reward, we compare it with an immediate reward via Q-value performance. Q-value performance is a judgment of whether a suits s. From Figure 6, we can find that both long-term and immediate reward would converge to a similar value, but the Q-value of immediate reward increases first and then decreases. The higher Q-value in the training procedure indicates that the immediate reward may get trapped in a local optimum. By contrast, the long-term reward allows our model to make decisions from a global perspective and the Q-value converge smoothly.



(a) Last.fm-Myspace (b) Linkedin-Aminer

Fig. 6 Q-value performance with different reward on Last.fm-Myspace and Linkedin-Arminer. The x-axis shows the training episodes. The y-axis shows the normalize Q-value of each session. The value of immediate reward is equal to $1/-1$, while long-term reward is $\frac{1}{t}/\frac{-1}{t}$

3.3.4 Effectiveness of RLink Components

This experiment is designed to validate the effectiveness of main components in the Actor network, including the input features, attention layer, LSTM layer, action transformer and decoder. We systematically eliminate the corresponding component and define the following variants of RLink.

- $RLink_{LINE}$ (RLink with LINE as pre-trained embedding method): This variant replace the pre-trained embedding method Node2vec by LINE [32] to evaluate the diversity of different pre-trained identity embeddings.
- RLink-FEED (RLink without feedback information): This variant is to evaluate the feedback fed into the Encoder. So, we just use the representation of matched identity pair as the history matching information.
- RLink-LSTM (RLink without LSTM layer): This variant is designed to evaluate the contribution of LSTM layer. We replace the LSTM layer by simple fully-connected layer.
- RLink-ATT (RLink without attention machine): This variant is designed to evaluate the contribution of the attention layer. So, we remove the attention machine after the LSTM layer.

Table 4 Performance on different RLink components

Datasets	Methods	Precision	Recall	F1
Last.fm-Myspace	RLink$_{LINE}$	0.8511	0.7732	0.8014
	RLink-FEED	0.7055	0.7182	0.7118
	RLink-LSTM	0.8014	0.7189	0.7579
	RLink-ATT	0.7921	0.7312	0.7604
	RLink-TRANS	0.7155	0.6367	0.6738
	RLink$_{MLP}$	0.8297	0.7354	0.7797
	RLink	**0.8571**	**0.7768**	**0.8038**
Aminer-Linkedin	RLink$_{LINE}$	0.9213	0.8987	0.9099
	RLink-FEED	0.8372	0.8265	0.8319
	RLink-LSTM	0.8648	0.8434	0.8540
	RLink-ATT	0.8726	0.8573	0.8649
	RLink-TRANS	0.8439	0.8191	0.8313
	RLink$_{MLP}$	0.9076	0.8879	0.8976
	RLink	**0.9285**	**0.9027**	**0.9154**

- RLink-TRANS (RLink without transformer layer): In this variant, we replace the action transformer component by fully-connected layer to evaluate its effectiveness.
- *RLink*$_{MLP}$ (RLink utilizes MLP as decoder): In this variant, we replace the single neural network (NN) layer in the decoder by MLP to evaluate the effective of NN layer.

The results are shown in Table 4. We can find that RLink$_{LINE}$ adapts LINE to pre-train user identity embedding vectors, which achieves similar performance to RLink. This phenomenon indicates that the pre-trained identity embeddings could not have great influence on the performance of UIL. RLink-FEED performs the worst among all variants, which demonstrates that the feedback information significantly promotes the performance of our model. RLink-LSTM performs worse than RLink, which suggests that capturing the long-term memory of the history matched information by LSTM is very beneficial for the subsequent matching. RLink-ATT verifies that incorporating attention mechanism can better capture the influence of each previous decision than only LSTM. RLink-TRANS verifies that action transformer could map a^{val} and a^{cur} exactly. RLink$_{MLP}$ verifies that simply neural network in the decoder phase could perform better than more complicated model (MLP). RLink outperforms all its variants, which indicates the effectiveness of each component for UIL.

4 Related Work

In this section, we briefly introduce previous works related to our study, including user identities linkage and reinforcement learning.

4.1 User Identities Linkage

Previous UIL works consider UIL as a matching problem, or utilize classification models and label propagation algorithms to tackle this task. Matching-based methods build a bipartite graph according to affinity score of each candidate pair of identities and achieves one-to-one matching for all user identity pairs based on this bipartite graph [24,25]. The basic principle is Stable Marriage Matching. Classification-based models classify whether each candidate matching is correct or not. Commonly used classifiers include Naive Bayes, Decision tree, Logistic regression, KNN, SVM and Probabilistic classifier[4, 21,20,33]. Label-Propagation based methods discover unknown user identity pairs in an iterative way from the seed identity pairs which have been matched[8,14,19,23,38]. Some recent works prefer to combine above methods, for example, [35] computes local consistency based on matching model and applies label propagation for global consistency.

With the development of network embedding and deep learning, embedding based methods and deep learning models have been utilized to solve UIL problem. Embedding based methods embed each identity into the low-dimensional space which preserve the structure of network firstly, and then align them via comparing the similarity between embedding vectors across networks [33,31,37,18]. However, those two-step methods need two subjects which are difficult to optimize. [11] proposes a unified framework to address this challenge, where the embeddings of multiple networks are learned simultaneously subject to hard and soft constraints on common users of the network. [39] introduces an active learning method to over the sparsity of labeled data, which utilizes the numerous unlabeled anchor links in model building. Finally, inspired by the recent successes of deep learning in different tasks, especially in automatic feature extraction and representation, [37] proposes a deep neural network based algorithm for UIL. LHNE [42] embeds cross-network structural and content information into a unified space by jointly capturing the friend-based and interest-based user co-occurrence in intra-network and inter-network, respectively. And then align users based on those embedding vectors. uStyle-uID[43] utilizes user writing and photo style to predict the label of identity pairs in Darknet network, while iDev[44] predict the label of identity pairs based on the feature coding published by user in public social coding platforms. DALAUP[45] utilizes active learning to solve the problem that labeled data is difficult to obtain in the UIL task.

4.2 Reinforcement Learning

Reinforcement Learning (RL) is one of the most important machine learning methods, which gets optimal policy through trail-and-error and interaction with dynamic environment. Generally, RL contains two categories: model-based and model-free. And the most frequently used method is model-free reinforcement learning methods which can be divided into three categories: Policy-

based RL, Q-learning and Actor-Critic. The policy-based RL [3,6] learns the policy directly which compute probability distribution of every action. The Q-learning [16,30] learns the value function which evaluate the value of each potential action-state pair. And Actor-Critic [36,29] learns the policy and value function simultaneously which aims to evaluate the quality of a deterministic action at each step.

Recently, because of various advantages of RL, RL has been successfully applied in many fields, such as Game [16,27], Computer Vision [40] and Natural Language Processing [3]. However, due to the complexity of online social network analysis tasks, works based on RL are less than in other fields. [28] is an early work based on reinforcement learning in social network which argues modeling network structure as dynamic increases realism without rendering the problem of analysis intractable. Existing methods utilize DQN or Q-learning framework, such as [7] applies DQN to address Graph Pattern Matching problem and [22] applies Q-Learning to search expert in social network. Inspired by the above works, we consider UIL as a markov decision problem and apply the reinforcement learning framework. However, computing Q-value of each identity pair is time-consuming. So, we adapt Actor-Critic framework to address UIL in this paper.

5 Conclusion

In this paper, we consider user identity linkage as a sequence decision problem and propose a reinforcement learning based model. Our model directly generates a deterministic action based on previous matched information via Actor-Critic framework. By utilizing the information of previously matched identities and the social network structures, we can optimize the linkage strategy from the global perspective. In experiments, we evaluate our method on Foursquare-Twitter, Last.fm-Mysapce and Linkedin-Aminer datasets, the results show that our method outperforms state-of-the-art solutions. There are many other information (e.g., user attribute information) in the network, and the links (e.g., such as AP (Author-Paper) and PC (Paper-Conference)) in the network are varied. Therefore, in the future, we would exploit those attribute information and varied links to further improve the UIL tasks.

Acknowledgements The authors would like to thank all the people who have contributed to user identity linkage archive for their selfless work. We also thank the anonymous reviewers for their valuable advice. The work is supported by the National Key R&D Program of China (NO.2018YFB1004704), the National Natural Science Foundation of China (U1736106).

References

1. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining (KDD), pp. 855-864 (2016)

2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: The International Conference on Learning Representations (ICLR) (2014)
3. Fang, Z., Cao, Y., Li, Q., Zhang, D., Zhang, Z., Liu, Y.: Joint Entity Linking with Deep Reinforcement Learning. In: The World Wide Web Conference (WWW), pp. 438-447 (2019)
4. Goga, O., Lei, H., Parthasarathi, S. H. K., Friedland, G., Sommer, R., Teixeira, R.: Exploiting innocuous activity for correlating users across sites. In: The World Wide Web Conference (WWW), pp. 447-458 (2013)
5. Kong, X., Zhang, J., Yu, P. S.: Inferring anchor links across multiple heterogeneous social networks. In: Proceedings of the ACM international conference on Information & Knowledge Management (CIKM), pp. 179-188 (2013)
6. Hu, M., Peng, Y., Huang, Z., Qiu, X., Wei, F., Zhou, M.: Reinforced mnemonic reader for machine reading comprehension. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 4099-4106 (2018)
7. Kanezashi, H., Suzumura, T., Garcia-Gasulla, D., Oh, M. H., Matsuoka, S.: Adaptive Pattern Matching with Reinforcement Learning for Dynamic Graphs. In: the IEEE International Conference on High Performance Computing (HiPC), pp. 92-101 (2018)
8. Korula, N., Lattanzi, S.: An efficient reconciliation algorithm for social networks. Proceedings of the VLDB Endowment 7(5), 377-388 (2014)
9. Lacoste-Julien, S., Palla, K., Davies, A., Kasneci, G., Graepel, T., Ghahramani, Z.: Sigma: Simple greedy matching for aligning large knowledge bases. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining (KDD), pp. 572-580 (2013)
10. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. In: The Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL), pp. 260-270 (2016)
11. Liu, L., Cheung, W. K., Li, X., Liao, L.: Aligning Users across Social Networks Using Network Embedding. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 1774-1780 (2016)
12. Arora, S., Liang, Y., Ma, T.: A simple but tough-to-beat baseline for sentence embeddings.In: The International Conference on Learning Representations (ICLR) (2017)
13. Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Sliver, D., Wierstra, D.: Continuous control with deep reinforcement learning. In: The International Conference on Learning Representations (ICLR) (2016)
14. Liu, S., Wang, S., Zhu, F., Zhang, J., Krishnan, R.: Hydra: Large-scale social identity linkage via heterogeneous behavior modeling. In: International Conference on Management of Data (SIGMOD), pp. 51-62 (2014)
15. Luong, M. T., Pham, H., Manning, C. D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1412-1421 (2015)
16. Lample, G., Chaplot, D. S.: Playing FPS games with deep reinforcement learning. In:Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pp.2140-2146 (2017)
17. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M.A., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Shanelegg.,Hassabis, D.: Human-level control through deep reinforcement learning. Nature 518(7540), 529-533 (2015)
18. Mu, X., Zhu, F., Lim, E. P., Xiao, J., Wang, J., Zhou, Z. H.: User identity linkage by latent user space modelling. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining (KDD), pp. 1775-1784 (2016)
19. Narayanan, A., Shmatikov, V.: De-anonymizing social networks. IEEE symposium on security and privacy 17(20), 173-187 (2009)
20. Goga, O., Perito, D., Lei, H., Teixeira, R., Sommer, R.: Large-scale correlation of accounts across social networks.University of California at Berkeley, Berkeley, California, Tech. Rep, TR-13-002 (2013).
21. Peled, O., Fire, M., Rokach, L., Elovici, Y.: Entity matching in online social networks. In: International Conference on Social Computing (SocialCom), pp. 339-344 (2013)

22. Peyravi, F., Derhami, V., Latif, A.: Reinforcement learning based search (RLS) algorithm in social networks. In: The International Symposium on Artificial Intelligence and Signal Processing (AISP), pp. 206-210 (2015)
23. Zafarani, R., Tang, L., Liu, H.: User identification across social media.ACM Transactions on Knowledge Discovery from Data (TKDD) 10(2), 1-30 (2015)
24. Riederer, C., Kim, Y., Chaintreau, A., Korula, N., Lattanzi, S.: Linking users across domains with location data: Theory and validation. In: The World Wide Web Conference (WWW), pp. 707-719 (2016)
25. Labitzke, S., Taranu, I., Hartenstein, H.: What your friends tell others about you: Low cost linkability of social network profiles. In: Proc. 5th International ACM Workshop on Social Network Mining and Analysis, pp. 1065-1070 (2011)
26. Shu, K., Wang, S., Tang, J., Zafarani, R., Liu, H.: User identity linkage across online social networks: A review. Acm Sigkdd Explorations Newsletter 18(2), 5-17(2017)
27. Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G.,Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., P.Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T.,Dieleman, S.: Mastering the game of Go with deep neural networks and tree search. Nature 529(7587), 484-489 (2016)
28. Skyrms, B., Pemantle, R.: A dynamic model of social network formation. In: Adaptive networks, pp. 231-251 (2009)
29. Sutton, R. S., Barto, A. G.:Reinforcement learning: An introduction. IEEE Trans. Neural Networks 9(5), 1054-1054 (1998)
30. Taghipour, N., Kardan, A.: A hybrid web recommender system based on q-learning. In: Proceedings of the 2008 ACM symposium on Applied computing (SAC), pp. 1164-1168 (2008)
31. Tan, S., Guan, Z., Cai, D., Qin, X., Bu, J., Chen, C.: Mapping users across networks by manifold alignment on hypergraph. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pp. 159-165 (2014)
32. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: The World Wide Web Conference (WWW), pp. 1067-1077 (2015)
33. Man, T., Shen, H., Liu, S., Jin, X., Cheng, X.: Predict anchor links across social networks via an embedding approach. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI),Vol. 16, pp. 1823-1829 (2016)
34. Zhang, J., Yu, P. S.: Pct: partial co-alignment of social networks. In: The World Wide Web Conference (WWW), pp. 749-759 (2016)
35. Zhang, Y., Tang, J., Yang, Z., Pei, J., Yu, P. S.: Cosnet: Connecting heterogeneous social networks with local and global consistency. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining (KDD), pp. 1485-1494 (2015)
36. Zhao, X., Xia, L., Zhang, L., Ding, Z., Yin, D., Tang, J.: Deep reinforcement learning for page-wise recommendations (RecSys). In: Proceedings of the 12th ACM Conference on Recommender Systems, pp. 95-103 (2018)
37. Zhou, F., Liu, L., Zhang, K., Trajcevski, G., Wu, J., Zhong, T.: Deeplink: A deep learning approach for user identity linkage. In: The IEEE Conference on Computer Communications (INFOCOM), pp. 1313-1321 (2018)
38. Zhou, X., Liang, X., Zhang, H., Ma, Y.: Cross-platform identification of anonymous identical users in multiple social media networks. IEEE transactions on knowledge and data engineering 28(2), 411-424 (2016)
39. Zhu, J., Zhang, J., Wu, Q., Jia, Y., Zhou, B., Wei, X., Yu, P. S.: Constrained active learning for anchor link prediction across multiple heterogeneous social networks. Sensors 17(8), 1786 (2017)
40. Zoph, B., Vasudevan, V., Shlens, J., Le, Q. V.: Learning transferable architectures for scalable image recognition. In:Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), pp. 8697-8710 (2018)
41. Shang, Y., Kang, Z., Cao, Y., Zhang, D., Li, Y., Li, Y., Liu, Y.: PAAE: A Unified Framework for Predicting Anchor Links with Adversarial Embedding. In: the IEEE International Conference on Multimedia and Expo (ICME), pp. 682-687 (2019)

42. Wang, Y., Feng, C., Chen, L., Yin, H., Guo, C., Chu, Y.: User identity linkage across social networks via linked heterogeneous network embedding. World Wide Web (WWWJ) 22(6), 2611-2632 (2019)
43. Zhang, Y., Fan, Y., Song, W., Hou, S., Ye, Y., Li, X., Wang, J., Xiong, Q.: Your style your identity: Leveraging writing and photography styles for drug trafficker identification in darknet markets over attributed heterogeneous information network. In: The World Wide Web Conference (WWW), pp. 3448-3454 (2019)
44. Fan, Y., Zhang, Y., Hou, S., Chen, L., Ye, Y., Shi, C., Zhao, L., Xu, S.: idev: Enhancing social coding security by cross-platform user identification between github and stack overflow. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 2272-2278 (2019)
45. Cheng, A., Zhou, C., Yang, H., Wu, J., Li, L., Tan, J., Guo, L.: Deep active learning for anchor user prediction. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pp. 2151-2157 (2019)