World Wide Web (2020) 23:75–101
https://doi.org/10.1007/s11280-019-00697-7

A content search method for security topics in microblog based on deep reinforcement learning

Nan Zhou[1] · Junping Du[1] · Xu Yao[1] · Wanqiu Cui[1] · Zhe Xue[1] · Meiyu Liang[1]

Received: 18 April 2018 / Revised: 16 April 2019 / Accepted: 23 May 2019
Published online: 28 July 2019

Abstract

Traditional methods treat the search problem as a process of selecting and ranking sequential documents. The methods have been proved effective and are widely used in the web search domain. However, due to the complexity and particularity of microblog text contents, the classical methods are rarely used microblog searches for specific topics. Focusing on the issue of searching for specific topics in microblog content, we present a microblog search method for security topics based on deep reinforcement learning by modeling the microblog search for specific topics as a continuous-state Markov decision process. We also design a novel deep Q network to evaluate the relevance of microblog content based on the target topic. We adopt reinforcement learning to solve the microblog search problem using an intelligent strategy and evaluate content relevance through deep learning. Experiments conducted on a real-world dataset show that our approach outperforms the selected baseline methods.

Keywords microblog search · reinforcement learning · deep Q networks · deep learning · information search

✉ Junping Du
junpingdu@126.com

Nan Zhou
zhounan345@163.com

Xu Yao
theoyao@163.com

Wanqiu Cui
wanqiu.wd@gmail.com

Zhe Xue
xuezhe@bupt.edu.cn

Meiyu Liang
meiyu-1210@126.com

Extended author information available on the last page of the article

1 Introduction

The microblog search problem has attracted researchers' interest. The emergence of smart mobile devices with integrated application modules has caused microblog data to explode on both traditional and mobile social network platforms. People share experiences, stories and funny content using short texts, pictures and brief videos through various microblog platforms [24]. Content posted on microblog platforms concerning security topics reflect people's views on public events, especially for disaster events. Social network platforms such as Twitter and Weibo are indispensable sources of microblog information. Public views on events—especially those concerning security—spread rapidly on these social network platforms. Searching for useful information about public events from microblog is becoming increasingly important because microblog information has become a vital source of news and life experiences.

Security topics in microblogs involve emergencies and disasters reported or discussed by users; otherwise, there is no difference between security topics and other topics from the standpoint of data features or statistical characteristics. However, from a social influence aspect, security topic information in microblogs spreads faster and wider. Disseminating security topic content on social networks can have wide-ranging social impacts [28, 47] that people with ulterior motives can capitalize on to create a panic. Thus, searching microblog content appropriately for target topics based on user-perceived utility is a hotspot in the social network information search domain. The challenge in searching microblog content for a specific security topic target is to find comprehensive sets of related content, which can be posed as an information search problem [38]. The term "user-perceived utility", or used as "user-perceived quality" which means "user-oriented system behavior" in interaction systems [15], has been proposed by Dolotta et al. [14]. We adopt the concept combined with information retrieval evaluations to assess the learning procedure of the proposed method.

Due to the limited length, casual expression and arbitrary writing that are characteristics of microblog messages, the sentiments of short texts posted on microblog are often ambiguous [17]. These attributes distinguish microblogging searches from traditional web searches. Traditional web searching and ranking operations rely on search engines that crawl and index web content efficiently. Search engines for applications in daily lives follow the ad hoc retrieval strategy [59] are aimed at providing satisfying search results for users. The high degree of content aliasing and the prevalence of semantic noise in microblog makes applying traditional methods to microblog search and ranking problems infeasible. Thus, microblog search and ranking tasks face several challenges, meanwhile, the advent of big data and deep learning techniques provide research opportunities. Given these circumstances, deep learning-based methods such as the Deep Structured Semantic Model (DSSM) [23], Convolutional Latent Semantic Model (CLSM) [44, 45] and others have been used to solve big data information search problems. In addition, neural click-through models [5] have been proposed for web search problems.

For search problems, matching and ranking are the key components. Wang et al. [51] proposed a listwise approach for ranking-oriented collaborative filtering. Guo et al. [19] presented a deep relevance matching model for the ranking problem. Methods for rank learning combined with deep neural networks [30, 43] have been proposed to solve the ranking problem for big data searches. However, these search strategies are not suitable for microblog data features. The conventional methods of information search divide the search procedure into two main parts: matching and ranking. Ranking is usually modeled as a static process that relies on similarity metrics; however, these types of methods over-rely on matching similarities.

In microblogs, the document form and limited content length result in extreme challenges for information retrieval [41]. Microblog search usually involves the latest news and events [12], especially for security related topics that reflect complex social attributes. The existing studies have focused mainly on searches for relevant information based on semantic similarity discrimination combined with the spatiotemporal characteristics of microblogs [34]. One difficulty of microblog search is representing the contents accurately given microblog sparsity issues. Researchers have approached microblog search based on short text retrieval methods. Hasanain and Elsayed [20] studied query performance prediction for microblog search based on short texts to estimate the effectiveness of microblog search systems in the absence of feedback. Agarwal et al. [1] adopted keyword search to find contextual messages from the short-text data streams in microblogs. They revealed that microblog search based on short text retrieval tends to result in only recent messages rather than the most relevant ones, which weakens the search utility perceived by users. Therefore, how to conduct microblog search while fully considering user-perceived utility is yet another challenge for microblog search.

In this paper, we present a content search method for specific topics (MDPMS) based on deep reinforcement learning that is specifically targeted toward searching for security topics in microblogs. A novel and dynamic content relevance evaluation strategy based on the Deep Q Network (DQN) is proposed. The proposed DQN structure is composed of convolutional neural networks (CNN) and long short-term memory (LSTM) [22]. CNN is an efficient neural network architecture for learning multidimensional representations from the data [48]. LSTM takes textual feature representations and outputs action-value calculations, which are used to dynamically evaluate the matching degree of diverse search results for the corresponding position during the content matching procedure. The proposed method follows a Markov decision processes (MDP), in which we regard each subsequence of the search results as an individual state. Standard reinforcement learning methods for MDP are applied to construct appropriate result sequences [35]. From an information search perspective, we calculate action values as the matching degrees for different ranking sequences. For complicated microblog data semantic features, we utilize the CNN to compact the local semantic features covered by specific words into low-dimensional semantic representations. LSTM, which is the key component of the proposed DQN, is adapted to calculate referential action values for each microblog semantic feature from the convolutional feature representation sequences. The action values are used to select an action: whether to include the corresponding microblog content in the ranked search results list.

The proposed method provides several advantages: 1) it dynamically evaluates the microblog content search matching degree by gauging how well the microblog content accords with corresponding topics; 2) it presents a novel DQN framework composed of a CNN and an LSTM to calculate the action values; 3) it realizes raw microblog content mapping to the most appropriate ranked content list in an end-to-end manner.

For the first advantage, the matching degrees are calculated in accordance with the corresponding ranking results. The resulting sequences change in length and content when new microblog content is selected as a result at the corresponding position. Different result sequences form different search states at each time step. The evaluations are trained by reinforcement learning while the states change cyclically by episodes. For the second advantage, reinforcement learning is applied to optimize the parameters for the designed DQN constructed by the CNN text feature representations and LSTM. The well-trained DQN structure addresses the microblog content features and outputs the matching degrees at corresponding positions dynamically during the changeable search state. Finally, existing methods calculate ranking scores directly for documents based on different queries [55] but ignore the user-perceived utility of information about different search states. Instead, the proposed method models the different action values as the user perceived utility following reinforcement learning, which makes the search ranking task more intelligent.

The rest of this article is organized as follows. In Section 2, we review related works about microblog search and discusses deep reinforcement learning applications for information search. In Section 3, the fundamentals of the proposed method are presented. Section 4 provides the details of content evaluation based on the DQN. Comprehensive experiments on real-world dataset are presented in Section 5, and Section 6 concludes this paper.

2 Related works

2.1 Microblog search

Microblog services have become increasingly popular since the advent of smart mobile devices [37]. In recent years, microblog retrieval [2, 7, 11] has attracted great attention from researchers worldwide. Zhang et al. [61] studied top-*k* disjunction query processing issues in microblog search. They proposed a compact indexing method named "judicious searching" for microblog search in a huge dataset. Basu et al. [3] proposed a microblog search method that used context-specific stemming to capture diverse microblog contents based on word embedding. Basu et al. [4] studied the issues of microblog search in disaster situations. Wang et al. [52] proposed a feedback concept model to solve the microblog retrieval problem using query expansion. The strategy of query expansion has also been adopted to solve real-time social network search problems. Zhang et al. [60] adopted a query-biased ranking model with a semi-supervised algorithm which they used to capture query characteristics. Xia et al. [56] solved the problem of complex query analysis by studying the top-*k* most significant temporal keywords.

Ranking is a key component of information search. Feng et al. [16] presented a reinforcement ranking model based on graphical emoticons as sentimental labels of microblog posts. Zhang et al. [62] presented a deep learning-based method to rank themes in microblog contents by calculating the similarities among visual features, text content and microblog popularity using a deep learning framework. Song et al. [49] provided a ranking algorithm for WeChat social network content by weighting the vector space of the entry position using document content matching technology.

De Maio et al. [13] introduced a method to readapt the ranking of preferred tweets (those more likely to be interesting to users). This method was based on deep learning through which a ranking model was built to measure tweets re-ranked from the top-ranked list.

2.2 Deep reinforcement learning applications on information search

Recently, deep reinforcement learning has been applied in many research fields, i.e., artificial intelligence, intelligent control, robots, and so on. Mnih et al. [35] developed a DQN that acted as a novel artificial agent to learn policies from high-dimensional sensory inputs. The agent parameterized an action-value function to play Atari games at a human performance level. Mnih et al. [36] introduced an asynchronous gradient descent method for reinforcement learning to optimize deep neural network controllers under a lightweight framework. Silver et al. [46] proposed an algorithm combining a Monte Carlo simulation with policy networks to optimize reinforcement learning from self-playing games. The Markov decision process [31, 39] is the core model of reinforcement learning and is also employed to construct ranking models. Xia et al. [50] utilized a continuous Markov decision process to construct a diverse ranking model. The model adopted a policy gradient reinforcing algorithm [8] to adjust the parameters to maximize the expected long-term discounted rewards.

Reinforcement learning strategies that follow the Markov decision process have been deployed to solve information search problems. Diversifying search result rankings [58] is an important goal in information search tasks. Xia et al. [55] formalized search ranking process as a set of sequential decision making processes that were further modeled as a continuous-state Markov decision process. The method learns to make decisions to choose which policies are used to rank documents in the corresponding positions based on search state. They used the policy gradient algorithm to train the method to earn an appropriate future reward. Keyhanipour et al. [25] applied reinforcement learning to research rank-aggregation. The method integrated data fusion and reinforcement learning algorithms such as Q learning [53] and SARSA [63] to obtain the best aggregated search result. Reinforcement learning frameworks are also used to address the challenges of high-dimensional training data. Click-through features were applied to reinforcement learning in [26]. Real-time web search [32] is another key aspect of information searching.

For large amount of data, deep representations have been shown to be indispensable elements for deep learning applications [21]. The related works concerning the use of deep reinforcement learning applications in information search show that the combination of deep representations and massive datasets provides opportunities for solving the problem of microblog searches for specific topics through deep reinforcement learning. Wei et al. [54] devised a learning to rank algorithm based on Markov decision process to calculate the ranking evaluations of all positions in the result list. The algorithm adopted policy gradient, an on-policy reinforcement learning algorithm, to realize reinforcement for the information search. In this paper, we proposed a microblog search method based on deep Q network (DQN) which is an off-policy reinforcement learning algorithm. Different from policy gradient, DQN selected appropriate actions to optimize a policy, whereas the policy gradient algorithm uses the policy to generate samples as same as the one used for updating parameters [50].

3 The proposed MDPMS method

We model the microblog search task as a continuous-state MDP [60]. For a microblog search task related to a specific security topic, we have the query content q and a set of microblog contents $\mathrm{M} = \{\mathrm{m}_1, \mathrm{m}_2, \ldots, \mathrm{m}_n\}$. We obtain the final ranking sequence $\mathrm{R} = \{\mathrm{r}_1, \mathrm{r}_2, \ldots, \mathrm{r}_l\}(l < n)$ through a supervised reinforcement learning procedure modeled by MDP. Each microblog content to be searched is mapped into a latent semantic space (m_n, $\mathrm{r}_l \in \mathrm{R}^d$) with the help of word embedding techniques, where each content representation is in a fixed length. The structure of the proposed method is illustrated in Figure 1.

The proposed method is primarily composed of a newly designed deep Q network (DQN). The DQN framework contains a pre-trained CNN framework to generate microblog content features and an LSTM with fully connected layers to generate action values. Microblog contents are selected or skipped as a search result based on the action values. The search result sequence states are recorded to provide rewards for training the proposed DQN. The details are presented in Sections 3.1 and 3.2. The proposed DQN is depicted and described in Section 4.

3.1 Formal definitions of the microblog searching and ranking procedures

Under MDP, microblog content searching and ranking for a specific security topic is formulated as a five-tuple <S, A, O, T, R>:

The state (S) stands for different search status states. Each ranking list has a corresponding state for different search statuses because S varies as the search results change. Briefly, we define state as a tuple for selected microblog contents as the search results and the encoded-user perceived utility from the selected contents.

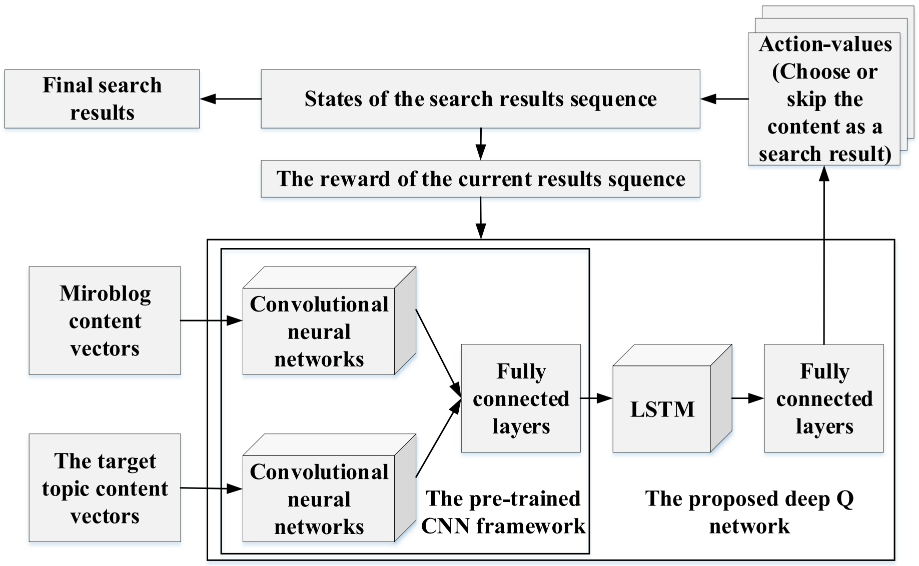

Final search results
States of the search results sequence
Action-values (Choose or skip the content as a search result)
The reward of the current results squence
Miroblog content vectors
Convolutional neural networks
Fully connected layers
LSTM
Fully connected layers
The target topic content vectors
Convolutional neural networks
The pre-trained CNN framework
The proposed deep Q network

Figure 1 The structure of the proposed microblog content-search method for security topics

S is designed as S = {D$_t$, P$_t$} at time step t, where D$_t$ is the result sequence of selected microblog contents in the preceding of t time steps. D$_t$ is defined as follows:

$$\mathrm{D}_t =< \mathrm{m}_1, \mathrm{m}_2, \dots, m_t >=< \mathrm{m}_{(n)} >_{n=1}^{t}, m_{(n)} \in \mathbb{R}^d \tag{1}$$

where m$_{(n)}$ is the ranked microblog content at position n. P$_t$ is the encoded user-perceived utility, like D$_t$, which is a variable length sequence defined as follows:

$$\mathrm{P}_t =< p_1, p_2, \dots, p_t >=< p_{(n)} >_{n=1}^{t}, p_{(n)} \in [-1, 1] \tag{2}$$

where $p_{(n)}$ is a scalar indicator of the user-perceived utility for the selected microblog content at the corresponding time step. In addition, m$_{(n)}$ and $p_{(n)}$ correspond to each other. At initialization, we define D$_t$ = ∅, P$_t$ = ∅ when t = 0.

Action (A) represents the actions to be selected for the corresponding microblog contents. For each microblog item, there are only two choices: choosing the content and including it in the results or skipping the content. At time step t, we define A as Act = {c, k}. In the action set, c represents choosing the content for state S$_t$ while k means skipping the content. For state S$_t$ at time step t, at = Act_DQN(S$_t$) determines whether the corresponding content m$_{t+1}$ to be selected to the result list or skipped at the current ranking position. The chosen microblog content is defined as C$_{\mathrm{m}(at=\mathrm{c})}$. As presented in the definition of S, the value of P$_t$ = DQN(C$_{\mathrm{m}(at=\mathrm{c})}$) acts as the action value [53] calculated by the DQN (details will be presented in next section).

Observation (O) is the observation of the current ranking environment. It is used to record the global status, including the state transformations and all the action choices. We define O as shown in Eq. (3):

$$\mathrm{O} =< \{\mathrm{S}_0, a_0\}, \{\mathrm{S}_1, a_1\}, \dots, \{\mathrm{S}_t, a_t\} >=< \left\{\mathrm{S}_{(n)}, a_{(n)}\right\} >_{n=1}^{t} \tag{3}$$

An observation is an expanded description of the state. The reinforcement procedure is reflected through observations.

A transition (T) is defined as the transformation of state S$_t$ at time step t to the next state, S$_{t+1,}$ activated by the actions selected by DQN with the θ parameters. The transition is a function presented as T:S × Act_DQN(S) → S as shown in Eq. (4):

$$\begin{aligned}\mathrm{S}_{t+1} &= \mathrm{T}(\mathrm{S}_t, \mathrm{Act\_}DQN(S_t;\theta)) \\ &= \mathrm{T}(< \mathrm{D}_t, \mathrm{P}_t >, at) \\ &= \begin{cases} \{< \mathrm{D}_t \oplus \mathrm{C}_{m(at=c)} >, \mathrm{P}_t \oplus \mathrm{DQN}\left(\mathrm{C}_{m(at=c)}\right)\} & \text{if } at = c \\ < \mathrm{D}_t, \mathrm{P}_t > & \text{if } at = k \end{cases}\end{aligned} \tag{4}$$

where ⊕ denotes a concatenation of sequences and elements. In the expression D$_t$ ⊕ C$_{m(at=c)}$, ⊕ concatenates the sequence of D$_t$ with the selected microblog content C$_{m(at=c)}$ to form the new sequence of state S$_{t+1}$ for the next time step.

A reward (R) is the evaluation of the quality of the search results for a training episode. The user-perceived utility defined in state is used to construct a reward in each episode. We adopt Normalized Discounted Cumulative Gain (NDCG) to formalize the reward representation. To acquire NDCG's properties, the user-perceived utility $P_t \in [-1, 1]$ at time step t is mapped to a relevance grade based on the domain segmentation (i.e., $rel_t = \text{switch\_map}(P_t)$) where rel_t is a positive integer that represents a relevance grade. The function switch_map(·) maps P_t to a corresponding relevance grade based on the domain segmentation. The reward is defined as r_NDCG as shown in Eq. (5), by applying the NDCG calculation:

$$\text{r\_NDCG}(P) = \frac{\text{switch\_}map(P_1) + \sum_{i=2}^{|P_t|} \frac{\text{switch\_}map(P_i)}{\log_2(i+1)}}{\sum_{i=1}^{|P_t|} \frac{2^{\text{switch\_}map(P_1)}-1}{\log_2(i+1)}} = \frac{rel_1 + \sum_{i=2}^{|P_t|} \frac{rel_i}{\log_2(i+1)}}{\sum_{i=1}^{|P_t|} \frac{2^{rel_i}-1}{\log_2(i+1)}}$$
$$\Leftrightarrow \frac{\sum_{i=1}^{p} \frac{2^{\text{switch\_}map(P_i)}-1}{\log_2(i+1)}}{\sum_{i=1}^{|P_t|} \frac{2^{\text{switch\_}map(P_i)}-1}{\log_2(i+1)}} = \frac{\sum_{i=1}^{p} \frac{2^{rel_i}-1}{\log_2(i+1)}}{\sum_{i=1}^{|P_t|} \frac{2^{rel_i}-1}{\log_2(i+1)}} \tag{5}$$

The reinforcement learning algorithm optimizes the model parameters under a supervised learning strategy using click-through labeled data of microblog contents. In this paper, the proposed method achieves reinforcement mainly by relying on continuous changes in the reward.

3.2 Reinforcement learning for microblog search

In this paper, we use the off-policy strategy to design a learning algorithm for search target topic content from microblogs. Due to the complicated microblog data characteristics that distinguish such data from traditional web data, we conduct a combined structured DQN model to process the semantic feature representations and the action-value relevance evaluations. The demand to learn better features than handcrafted ones motivated us to connect reinforcement learning algorithms to deep learning methods, which are applied to operate directly on high-dimensional microblog semantic features.

Inspired by reinforcement learning [50], the algorithm for the proposed method is shown in Algorithm 1. As an off-policy strategy, the proposed method operates following deep Q learning with experience replay [35], interacting with the microblog contents with click-through labels (i.e., $\{<d_1, L_1>, <d_2, L_2>, \ldots, <d_l, L_l>\}$), where l is the length of the microblog contents to be searched.

Setting the immediate reward for the current time step is an essential component of Algorithm 1. For each time step, the result list is rebuilt with the selected content. The immediate reward is calculated by Eq. (5), while the final reward of all the procedure is presented as Step 16 in Algorithm 1. We set $r_{t=0} = 0$ for initialization, while r_t represents the final reward of the time step t. Generating action-choosing scores known as "action values" presented by the probability distributions for choosing the actions is a vital step to realize the microblog searching and ranking during the MDP. The target of the algorithm is to learn parameters for the DQN to generate action values as user-perceived utilities to select related microblog contents. Algorithm 1 is the key to learning an intelligent ranking model for microblog searching. For learning purposes, we adopt the mean-square error function as the loss function, as shown as Eq. (6), following the standard reinforcement learning process:

$$\begin{aligned} L(\theta) &= \mathrm{E}\left[\left(\mathrm{Q}^*(S;\theta) - \mathrm{Q}(S;\theta)\right)^2\right] \\ &= \mathrm{E}\left[\left(r + \gamma \max\_a(\mathrm{Act\_DQN}(S;\theta)) - \mathrm{Q}(S;\theta)\right)^2\right] \end{aligned} \tag{6}$$

Algorithm 1 MDPMS learning

Input: The query microblog content q for the target topic, the data set of microblog contents

Output: Optimized ranking model, optimized DQN model with appropriate parameters θ

```
Step 1: Initialize the greed coefficient $\varepsilon$=0.01, the greed increment coefficient $ic$=1.001, and the
        greed threshold $gt \in (0, 1)$ as a random float value. The algorithm learning rate is $\eta$=0.1, and
        the discount coefficient for reward $\gamma$=0.1.
Step 2: Initialize the DQN model with the initial parameters $\theta_{t=0}$, episode=0
Step 3: While not converged:
Step 4:     Initialize State S={$D_t$, $P_t$}, Observation O=<∅>, episode+=1
Step 5:     For time step $t$=1 to T:
Step 6:         If $\varepsilon < gt$:
Step 7:             Choose a random action $at$
Step 8:         Else:
Step 9:             Choose an action $at$=max_$a$(Act_DQN ($S_t$; $\theta$))
Step 10:         If $\varepsilon < 0.9$: Increase the greed coefficient $\varepsilon = \varepsilon \times ic$
Step 11:        Randomly update the greed threshold $gt$ from the domain of (0, 1) as a float value
Step 12:        Execute the chosen action $at$
Step 13:        Execute Transition $S_{t+1}$=T($S_t$, Act_DQN ($S_t$; $\theta$))
Step 14:        Update State and Observation for the chosen action and the new state $S_{t+1}$
Step 15:        Rank results in State and Observation according to $P_t$
Step 16:        Set the immediate reward for the current time step $t$
                $reward = \begin{cases} r_t & \text{if terminal at time step } t+1 \\ r_t + \gamma \mathrm{r\_NDCG}(P_t \oplus P_{t+1}) & \text{if not terminal at time step } t+1 \end{cases}$
                $P_{t+1} = \begin{cases} \varnothing & \text{if skipped} \\ \mathrm{Act\_DQN}(S_{t+1};\theta) & \text{if chosen} \end{cases}$
Step 17:        Perform a gradient descent step on $(reward\text{-}\mathrm{r\_NDCG}(P_t \oplus P_{t+1}))^2$ according to Eq.(6)
Step 18:        Update the parameters of DQN along the learning procedure
```

The mean-square error loss function evaluates the mathematical expectation of the deviations between target values and real values. In the loss function, the target is Q*(S;θ), which is represented as $r_t + \gamma \mathrm{r\_NDCG}(P_t \oplus P_{t+1})$. In the expression, γ is the discounted factor controlling the target value change. Q(S;θ) outputs real values through which the target expression is used to estimate the action-value function.

The DQN is an action-value generator. The action values are used as the user-perceived utilities. Another key function of DQN is to analyze the semantic features of different microblog contents and queries for specific topic content. The proposed DQN combined with CNN and LSTM is described in Section 4.

4 Selecting and evaluating relevant contents based on DQN

The existing information-search methods depend on handcrafted relevance features for searching and ranking [35]. For these methods, feature quality directly determines the reliability. Furthermore, the existing methods calculate relevance scores by relying on interaction measures between the queries and documents at separate and fixed positions. The calculation procedure is regarded as a static process that neglects the effects of dynamically constructing the sub-ranking results in the final ranking result. The search method presented in Section 3 is intended to solve that problem.

As a search problem, evaluating content to obtain relevance scores is indispensable—especially for microblog contents with complex data characteristics. In this section, we propose a DQN functional framework, as shown in Eq. (4) and line 9, 13 in Algorithm 1, to process microblog semantic features under a semantic embedding space.

4.1 CNN and LSTM-based DQN to select relevant content

The typical microblog content characteristics are limited length, casual expression, and arbitrary writing. These features make searching microblog content for topics different from searching traditional web content because microblog content is mixed with large amounts of global semantic noise. To search microblog content for a specific topic, the contents are processed based on local semantic features expressed by representative words.

As a preprocessing step, word segmentation is conducted on the microblog contents and stop words are removed. In this process, the contents of a microblog $\mathbf{m}$ is modeled as a multidimensional vector $\mathbf{m} = <\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_p>$ using word embedding where $\mathbf{w}_p \in R^{wd}$ is a fixed-dimensional word vector. We used a pretrained Word2vec algorithm [33] to create the fixed-dimensional word vectors. The length of each microblog vector is also a fixed value that is greater than the number of words in the microblog content without stop words.

To learn the local semantic features of microblog contents, we utilize the CNN framework to perform convolutional and pooling calculations and generate compact representations. The LSTM framework with two fully connected layers is then applied to calculate the action value. Finally, the action value is used to evaluate the selected microblog content for the corresponding position in accordance with the query. The proposed DQN framework is shown in Figure 2.

As depicted in Figure 2, the CNN framework is deployed as the interface to extract the local semantic feature representations. The CNN is trained in pairwise fashion so that it is sensitive to the target topic content vectors. In the search process, the target topic contents are the corresponding queries from which the vectors are generated by the word embedding techniques. From a representation aspect, the query contents as the search target are preprocessed into the computable vectors at a uniform size in the same way as other microblog contents, as shown in Figure 2.

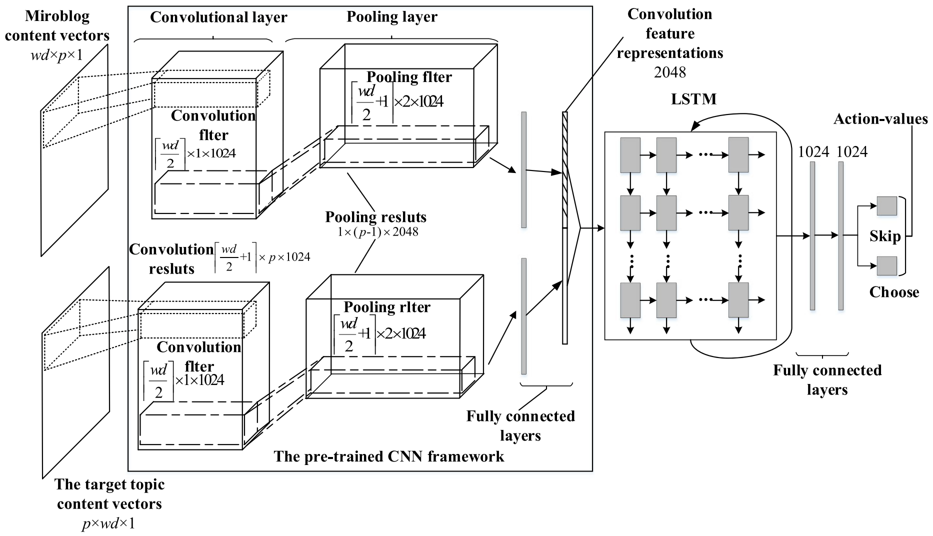


Figure 2 The proposed DQN framework

As stated earlier, the microblog contents are mapped into fixed-size vectors of $wd \times p \times 1$ where wd is the length of the word vectors constructing the p microblog vectors. Similar to image processing, the convolutions operate at a region of the microblog vectors followed by pooling conducted on the convolution result vectors. We perform the convolution computations at a size of $\lceil wd/2 \rceil \times 1$ (e.g., half the length of a word vector). The pooling layers have a size of $\lceil wd/2 + 1 \rceil \times 2$ and operate on the region of the convolution results. The convolutional layers and the pooling layers collaboratively calculate the semantic features of the microblog vectors. The convolutional feature representations, which involve both microblog content features and target topic content features, are generated by nonlinear transformations from the pooling result vectors. More computational details are presented in the following section. The compact feature representation sequences output by the CNN framework are input to the LSTM framework to calculate the action values. Hence, both the semantic and temporal dependencies of the content stream are considered in the LSTM. This LSTM evaluates whether the current content should be selected as a search result. The input sequence for the LSTM is composed of a series of contents that are evaluated to construct a continuous semantic and temporal state. The values output by the LSTM framework by the two fully connected layers are treated as the action values for the end stage of the input sequence. At initialization, the input sequence of the LSTM is padded with zero vectors; the sequence will be filled up by the accumulated content vectors over the time steps.

4.2 Computational details of the DQN

The aim of the convolutional layer is to extract the local semantic features contained by the representative word vectors. The convolutional filter has a size of $\lceil wd/2 \rceil \times 1$ in the first two dimensions and 1024 channels in the third dimension. The convolution filter covers half a word vector during each computation step, constructing the convolution result vector at a size of $\lceil (wd\text{-}(wd/2)/2 + 1)/1 \rceil \times 1 \times 1024$, as $\lceil wd/2 + 1 \rceil \times 1 \times 1024$.

More formally, we define the convolution operation as * between the content vector $\mathbf{V}_c$ and the convolution filter $\mathbf{F}$. Following Severyn's work [45], the convolution operation is defined as shown in Eq. (7).

$$\mathbf{S}_{\mathrm{cr}} = \sum_{i=0}^{i+\lceil wd/2\rceil-1} \sum_{j=0}^{j+p-1} \mathbf{V}_{\mathrm{c}\left[i:i+\left\lceil\frac{wd}{2}\right\rceil-1,j:j+p-1\right]} * \mathbf{F} \tag{7}$$

where $\mathbf{S}_{\mathrm{cr}}$ is the convolution result vector. The convolution filter covers the content vector $\mathbf{V}_c$ at the range of $[i{:}i+\lceil wd/2\rceil\text{-}1, j{:}j+p\text{-}1]$ (from i to $i+\lceil wd/2\rceil\text{-}1$ at the first dimension and from j to $j+p\text{-}1$ at the second dimension).

The convolution result vectors are passed to the activation function to generate the input of the pooling layer. The pooling layer aggregates the information and reduces the representation. The pooling operation is defined as follows:

$$\mathbf{S}_{\mathrm{pr}} = \mathrm{max\_pool} \sum_{i=0}^{i+\lceil wd/2\rceil} \sum_{j=0}^{j+p-2} \left(\mathrm{ReLU}\left(\mathbf{S}_{\mathrm{cr}\left[i:i+\left\lceil\frac{wd}{2}\right\rceil,j:j+p-2\right]} + \mathbf{b}_{ij}\right)\right) \tag{8}$$

where $\mathbf{S}_{\mathrm{pr}}$ is the pooling result vector. ReLU is used as the activation function and $\mathbf{b}_{ij}$ is the corresponding bias. We use max-pooling to realize the pooling operation at the range of $[i{:}i+\lceil wd/2\rceil, j{:}j+p\text{-}2]$ (from i to $i+\lceil wd/2\rceil$ at the first dimension and from j to $j+p\text{-}2$ at the second dimension).

The pooling filter has a size of $\lceil wd/2+1\rceil \times 2$ at the first two dimensions and 2048 channels at the third dimension. The filter covers generate the result vector at a size of $\lceil wd/2+1\text{-}(wd/2+1+1)/2\rceil \times (p\text{-}2+1) \times 1024$, or $1 \times (p\text{-}1) \times 2048$.

In the learning phase, we use a pairwise method to analyze the target contents together with the microblog contents to cause the CNN to be sensitive to the target representation vectors. The resulting generated convolution feature representations $\mathbf{R}_{\mathrm{cf}}$ are associated with both the microblog content features and target topic content features. The CNN utilizes the local perception properties to process the embedded semantic vectors of microblog contents and target topic contents. The local perception property of the CNN meets the special demands of microblog data characteristics to capture local semantic features among the large amounts of semantic noise. This supervised learning process generates the convolutional feature representations of microblog and target topic contents. To improve the coherence of the semantic information, an LSTM with two fully connected layers is used to analyze the large amounts of convolution feature representations. The goal of the LSTM is to yield an appropriate action value to guide the reinforcement learning algorithm to make a proper action choice.

LSTM is a Recurrent Neural Network (RNN) structure whose blocks are composed of a cell, an input gate, an output gate and a forget gate. The cell obtains new input information each time when the input gate i_t is activated at time step t. The final state h_t receives the latest cell output c_t when the output gate o_t is on. The forget gate is activated when the previous cell output $c_{t\text{-}1}$ should be forgotten. Under this strategy, the gradient will be trapped in the cell and prevented from vanishing too quickly [57]. The convolution features $\mathbf{R}_{\mathrm{cf}}$ generated by the

CNN are input into the LSTM in sequence. In this paper, we follow the formulation of Graves' work [18] to present the model shown in Eq. (9).

$$\begin{aligned}
i_t &= \sigma\left(\mathbf{W}_{\mathrm{R}i}\mathbf{R}_{\mathrm{cf}\_t} + \mathbf{W}_{hi}h_{t-1} + \mathbf{W}_{ci}\circ c_{t-1} + \mathbf{b}_i\right)\\
f_t &= \sigma\left(\mathbf{W}_{\mathrm{R}f}\mathbf{R}_{\mathrm{cf}\_t} + \mathbf{W}_{hf}h_{t-1} + \mathbf{W}_{cf}\circ c_{t-1} + \mathbf{b}_f\right)\\
c_t &= f_t\circ c_{t-1} + i_t\circ\tanh\left(\mathbf{W}_{\mathrm{R}c}\mathbf{R}_{\mathrm{cf}\_t} + \mathbf{W}_{hf}h_{t-1} + \mathbf{b}_c\right)\\
o_t &= \sigma\left(\mathbf{W}_{\mathrm{R}o}\mathbf{R}_{\mathrm{cf}\_t} + \mathbf{W}_{ho}h_{t-1} + \mathbf{W}_{co}\circ c_t + \mathbf{b}_o\right)\\
h_t &= o_t\circ\tanh(c_t)
\end{aligned} \tag{9}$$

where σ is the logistic sigmoid function, ° denotes the Hadamard product, and i_t, f_t, c_t, o_t, and h_t are the status values of input gate, forget gate, cell state, output gate and final state, respectively at time step t. $\mathbf{R}_{cf\_t}$ is the convolution feature input into the LSTM at time step t and **W** and **b** are the weight and bias parameters for the corresponding components of LSTM. LSTM is connected with two fully connected layers. The output of the LSTM, $\mathbf{l}_\mathrm{r}$, is input into these two layers to generate LSTM features $\mathbf{l}_\mathrm{fc}$. The action value is generated by Softmax as values distributions for different actions formatted as Eq. (10).

$$\begin{aligned}
\mathbf{l}_\mathrm{fc} &= \mathrm{ReLU}\left(\mathbf{W}_\mathrm{fc}{}'\mathbf{l}_\mathrm{r} + \mathbf{b}_\mathrm{fc}{}'\right)\\
Action\_value &= \mathrm{Softmax}\left(\mathbf{W}_\mathrm{fc}\mathbf{l}_\mathrm{fc} + \mathbf{b}_\mathrm{fc}\right)
\end{aligned} \tag{10}$$

where ReLU is the activation function used in Eq. (8), $\mathbf{W}_\mathrm{fc}{}'$ and $\mathbf{b}_\mathrm{fc}{}'$ are the weight and bias parameters in the first fully connected layer—similar to the $\mathbf{W}_\mathrm{fc}$ and $\mathbf{b}_\mathrm{fc}$ in the second layer.

We use softmax as the output of the fully connected layers to obtain the choice probability distribution after the two fully connected layers. The final output of the associated framework applies the values of the function of $Act\_DQN(\cdot)$ as the action-value distribution for "Choose" and "Skip."

5 Experiments and analysis

We conducted experiments to evaluate the proposed method of MDPMS on the real-world dataset from Sina Weibo. We selected four security topics as the specific search topics. The performances of the state-of-art information search methods based on traditional techniques and deep neural networks are evaluated for searching security topics.

5.1 Dataset

To search security topics in microblog content, we collected a dataset from Sina Weibo covering from June 10th, 2012, to September 7, 2016, containing 385,712 posts that included both relevant content and non-relevant content regarding the four selected security topics: Kunming terrorist attacks, Tianjin explosions, rainstorms in Hubei and fake vaccines in China. The proposed MDPMS method is trained through supervised learning with labeled data. The data—including the noise—of the security-related topics is randomly divided, with 70% used for training (including testing) and 30% used for validation. The statistics for the relevant contents as positive samples of these four events are shown in Table 1.

Table 1 Statistics of four events of microblogs

Security topics	Total number	Training number	Validation number
Kunming terrorist attacks	32,751	22,926	9825
Tianjin explosions	53,749	37,624	16,125
Rainstorms in Hubei	36,632	25,642	10,990
Fake vaccine in China	39,014	27,310	11,704

We split the contents of the four security topics into a training set and a validation set. The purpose is to ensure that the datasets include different specific topics that can reflect the commonality of security topics. The model is trained to evaluate common local semantic feature representations for the target topics, and the training set and the validation set both contain instances of all four specific security topic contents. The training procedure is conducted under supervised learning with labeled data by click-through.

The procedure of labeling the ranked position of the dataset is as follows. At first, the traditional query-likelihood language model [42] is used to simulate the general query process of a user to select positive samples from the dataset. The simulation is also used to acquire target contents and intentions of users under the hypothetical situation. The labeled data will be further recognized during the simulated situation, in which the action values represent the user-perceived utility. Then, we repeat the process 150 times to get the ranked positons of the contents in the result list. Among the results, the ranked position labels for the contents whose ranked positions are not changed are determined. For the remaining labels of the ranked positions, we manually adjust them to get the appropriate labels.

Following the click-through [44] of information retrieval, we labeled the dataset manually based on the results of queries using multiple keywords that are representative of the corresponding security topics. We also created sublabels for the dataset at different relevant levels in conjunction with the semimanual labels to improve the convenience of further evaluations. The relevance of these labels is determined during the training phase. Further evaluations are made using NDCG and MAP.

5.2 Experimental settings

MDPMS is based on an off-policy reinforcement learning algorithm. We initialized the algorithm by assigning parameters including the greed coefficient ε, the greed increment coefficient *ic*, the discount coefficient for reward γ and the parameter learning rate [29] η. The initial parameter values are shown in Table 2.

From Table 2, the greed coefficient ε controls how the algorithm select actions based on experiences. Initially, the algorithm needs to explore every possible action for different contents to build experience when it has no prior experience to rely on. This is why the greed

Table 2 Initializations of parameters

Parameters	Initial values
The greed coefficient ε	0.01
The greed increment coefficient *ic*	1.001
The discount coefficient for reward γ	0.1
The learning rate η	0.01

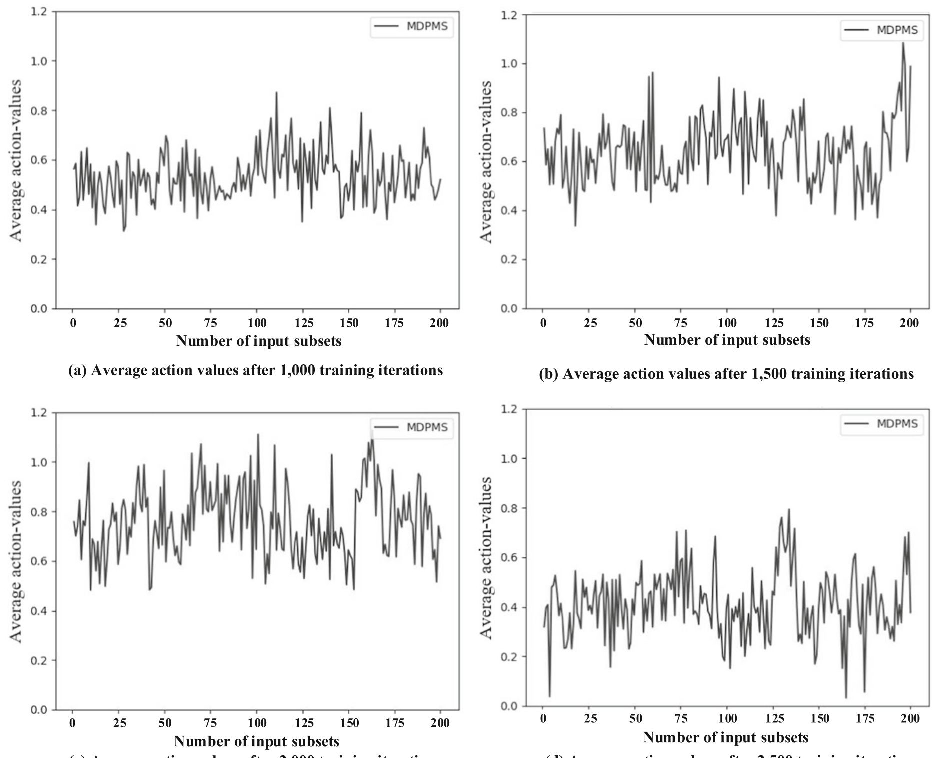
coefficient is initially set to $\varepsilon = 0.01$—to ensure that the algorithm is not greedy at initialization. The algorithm learns how to select suitable actions gradually for different microblog contents to gain higher rewards. This incremental greed process is controlled by the greed increment coefficient. The algorithm becomes greedier as the learning process progresses. The greed coefficient is updated by $\varepsilon = \varepsilon \times ic$ at each time step. The value range of the greed coefficient ε is a float value ranging from 0.01 to 0.9; the 0.9 limit ensures that the algorithm will never become totally greedy. The reward is updated when a new chosen content is selected and placed in the result list. The changing reward makes the algorithm more intelligent, helping it to know which action should be chosen when faced with different contents. However, the algorithm selects an action for the upcoming content under a greedy strategy: eventually, there is only a 10% chance that it will explore new possibilities for similar content to update its older experiences. This situation reflects the fact that its initial experiences are more valuable. The discount coefficient for reward is intended to balance the changing experience referential value shown in Eq. (6) and in Line 16 in Algorithm 1. The learning rate is a traditional concept in machine learning that controls the neural network parameter updating.

The algorithm is reinforced in accordance with the evaluations of action values and rewards. During the learning stage, the rewards are the feedbacks from the phased results of all the different episodes, from which the algorithm adjusts the action-selecting strategy for various microblog contents to obtain a better future reward. In each time step, an action value is generated to evaluate the action selected for a corresponding microblog content. This is the algorithm interface that analyzes the semantic features. The final outcome of the analysis process is conducted by DQN framework. The pretrained Word2vec generates a 60-dimensional vector for each word. The fixed-size microblog vector is a multiple-dimensional vector composed of word vectors; the first dimension of a microblog vector depends on the number of word vectors with stop-words removed. We define the microblog vector, which has a size of 60×100, as $wd = 60$, $p = 100$ in Figure 2, where 100 is larger than the word-count value of the longest microblog entry. Another key component is the LSTM, which receives sequences of convolution feature representations and calculates action values.

5.3 Parameter sensitivity experiments

We conduct experiments to verify the effectiveness of the proposed method. The search goal is to select related microblog entries by relying on the action selected for different entries. As presented in section 3, the actions are defined as “choose” and “skip.” The algorithm selects the appropriate action for different entries to determine which microblog contents should be included in the search results; thus, the actions selected for different specific entries directly determine the search results. We randomly selected 2000 microblog entries from the dataset to evaluate the changing process of choosing the actions. The 2000 posts were divided into 200-item mini-batch subsets. Each subset contains 10 posts to verify the average action values at different training phases under the greedy mechanism, and each content subset was unique. To present the results intuitively, we redefined the action value as a binary value (1 or -1), where the “choose” action-value is 1 and the “skip” action value is -1. The average action-value curves during training are shown in Figure 3.

The subsets are input in sequence; then, the action-selection trend for different subsets over a given set of time steps is shown in Figure 3. The reinforcement is reflected in the procedure used to gain action-selection experience to select different microblog posts as the search



(a) Average action values after 1,000 training iterations

(b) Average action values after 1,500 training iterations

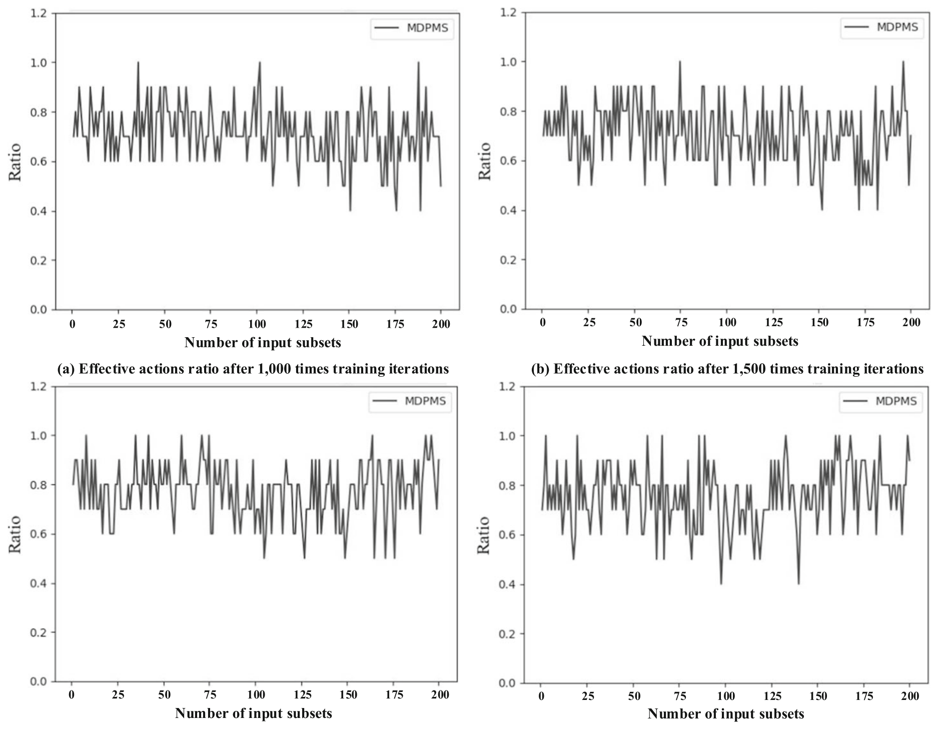
(c) Average action values after 2,000 training iterations

(d) Average action values after 2,500 training iterations

Figure 3 Average action values at different training phases

results. We use subsets of the selected samples to show the changes in the average action values for different subsets evaluated in sequence. As shown in Figure 3, we selected four training phases to demonstrate how the average action values change for different subsets. The subsets are input as a sequence. The action-value distributions on some subsets of the four training phases fall into the positive interval, meaning that overall, the algorithm selected "choose" more often than "skip.". Furthermore, the proportion of "choose" actions begins an upward trend between 1000 and 1500 training iterations for the 2000 content items. In contrast, a significant decline occurs after 2000 training iterations. This trend shows that the algorithm becomes greedy and stops exploring different possible actions on the same content. The fact that algorithm becomes greedy means that the greed coefficient has degenerated to a fixed-value. However, the algorithm parameters tend to converge at approximately 2500 training iterations. As presented in (a) to (c), the fluctuation center changes become stronger. After 2500 training iterations, the fluctuation center changes becomes similar to that after 2000 training iterations, showing that the training tends to converge.

We also evaluated the efficiency of the action selections based on the randomly selected contents as shown in Figure 4. The four images in Figure 4 present changes in the effective action ratios based on the input of different subset sequences. The effective actions mean choosing instead of skipping the correct contents should be picked up for the corresponding ranking positions. Intuitively, The more effective actions the higher user-perceived search utility. A series of effective actions make up the appropriate result list. Figure 4 shows the effectiveness of the proposed method, where we calculated the ratio of effective actions in all



(a) Effective actions ratio after 1,000 times training iterations

(b) Effective actions ratio after 1,500 times training iterations

(c) Effective actions ratio after 2,000 times training iterations

(d) Effective actions ratio after 2,500 times training iterations

Figure 4 Effective actions ratios at different training phases

the actions the algorithm made for constructing the search result list from different training phrases. Over the time steps for the training process, the actions selected by the algorithm for different subsets of microblog contents tend to be reasonable and accordingly, the effective actions ratio (the ratio of appropriate actions) changes as well.

Figure 4 shows the changing process of the effective actions ratio for different subsets input as a sequence. In Figure 4 (a) and (b), the curves fluctuate mainly within a range of 0.6–0.8. In contrast, the curves in Figure 4 (c) and (d) fluctuate primarily within a range of 0.7–0.9. There is an approximate 0.1-unit increase between 1000 and 2500 training iterations, while a stable trend appears from 2000 to 2500 training iterations. The stable curve trend confirms that the algorithm tends to converge after approximately 2500 training iterations. The increasing trend of the effective actions ratio demonstrates that the algorithm updates its parameters effectively to take appropriate actions for corresponding microblog entries.

As shown in Figures 3 (c), (d) and 4 (c), (d), the algorithm "skips" more content from 2000 to 2500 training iterations as the effective actions ratio stabilizes. This situation indicates that the algorithm has learned relatively suitable parameters for selecting the appropriate actions for corresponding content. The phenomenon presented in Figures 3 and 4 shows that the algorithm is sensitive to the semantic features of the security topics when selecting the most related contents as the search results.

We computed the loss values to verify the learning efficiency of the reinforcement process, as shown in Figure 5. Different from traditional learning methods, an increasing phase occurs initially. Under the greedy mechanism, the reinforcement learning algorithm gains action-selection experience by exploring all the actions for different content items under the control of

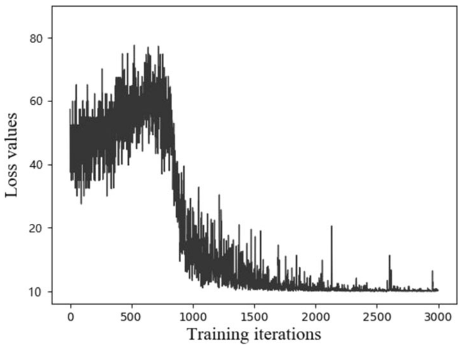


Figure 5 Effective action ratios at different training phases

the greed coefficient. The loss values curve of shows the changing trends of the average action values and effective actions ratios at different training phases. The loss begins to decrease at approximately 700 iterations and converges after 2000 iterations.

In the next sections, evaluations on the ranked search results lists are presented to verify the effectiveness of the proposed method.

5.4 Microblog searching results and analysis

The proposed MDPMS method was trained under supervised learning by numerous labeled microblog contents. In experiments, we selected 100 queries from the four specific security topics (25 from each) to verify the searching efficiency of the proposed method. The search experiments were conducted on the verification dataset combined with the four topics to verify the universality of the method used for identifying general security-topic content from microblogs. The different queries result in different ranked search result lists. The evaluations are operated on the average values of the evaluation metrics calculated from the search results for different values.

We adopt Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP) [57] as metrics to evaluate the search results for the top *n* ranking (e.g., NDCG@*n* and MAP@*n*). NDCG is calculated as shown in Eq. (5).

The MAP is calculated as shown in Eqs. (11) and (12).

$$AveP = \frac{1}{n}\sum_{q}(P@n \times r) \tag{11}$$

$$MAP = \frac{\sum_{q=1}^{|Q|} AveP(q)}{|Q|} \tag{12}$$

where r is the relevance score assigned to the content at position n with respect to a given query q.

We also conducted comparative experiments to verify the efficiency of the proposed method. We compared MDPMS with state-of-the-art methods for searching information, including BM25, Aho–Corasick DSSM, CLSM, RankNet and ListNet.

BM25 [40] uses a set of functions based on the bag-of-words model that ranks content based on query terms.

Aho–Corasick [10] is a type of dictionary-matching algorithm that locates elements of a finite set of strings.

RankNet [6] is a neural network learning-to-rank model for the underlying ranking trained by a probabilistic loss function using gradient descent.

ListNet [9] is a listwise learning-to-rank model for information search based on permutation probability and top n probability.

DSSM [23] is a deep neural network model that represents text strings in a latent semantic space and calculates the semantic similarities between queries and content.

CLSM [44] is a latent semantic model that incorporates convolution network structures over word vectors to find similarities between search queries and content.

MDPRank [21] is a learning-to-rank model for information search on the basis of MDP.

As listed in Table 3, the performances of the proposed MDPMS method and the baseline methods are evaluated using the NDCG@5, NDCG@10, NDCG@15 and NDCG@20 metrics on their average values. NDCG is applied to define the reward following the Markov decision process, which is the key factor in the method's performance. The average NDCG@n evaluates different levels of relevance degrees for the search results. We output the rewards as the four metrics for the top 5, top 10, top 15 and top 20 search results of the validation experiments and calculated the four values for the baseline methods in comparative experiments using the same validation set.

Table 3 shows a comparison of the search result metrics for the top 5, top 10, top 15 and top 20 results. Our method outperformed all the selected baseline methods. MDPMS is trained to achieve a high evaluation value on NDCG during the training phase, where the selected actions for different microblog contents guide it to construct an appropriate search result to gain a high reward. The reward is defined by integrating NDCG, through which MDPMS is validated with the goal to match the training level with the NDCG. In contrast, the baseline methods produce search results from a static procedure based on similarity functions or a pretrained model. In particular, the BM25 model is a representative information search method used in the traditional web search domain that is based on Term Frequency-Inverse Document Frequency (TF-IDF); however, this property is not well-matched with the characteristics of microblogs, which contain casual expressions and arbitrary writings.

We also used MAP@n to evaluate the average Precision@n values of relevant content in the result lists. These values are shown in Table 4.

Table 3 Comparison of MDPMS and baseline methods on average NDCG@n

	NDCG			
	@5	@10	@15	@20
BM25	0.599	0.631	0.690	0.658
Aho–Corasick	0.589	0.669	0.662	0.668
RankNet	0.649	0.668	0.671	0.683
ListNet	0.683	0.688	0.694	0.699
DSSM	0.640	0.653	0.671	0.662
CLSM	0.662	0.671	0.700	0.702
MDPRank	0.691	0.706	0.725	0.724
MDPMS	0.716	0.728	0.741	0.752

As shown in Table 4, MDPMS outperforms the selected baseline methods for all the average values (MAP@5, MAP@10, MAP@15 and MAP@20). The definition of the MAP calculation in Eq. (12) and Eq. (13) demonstrate a property of MAP: the higher the rank of the relevant content is, the higher the MAP values are. The values of the average NDCG@5, Precision@5 and MAP@5 of MDPMS indicate that the proposed method has advantages in relevance ranking for security topic content searches in microblogs. The overall evaluations demonstrate that the proposed MDPMS method performs better than the selected baseline methods for security topic content searches in microblog. The baseline methods stem from traditional web search methods, learning-to-rank methods and deep learning methods. BM25 is the representative web search method based on TF-IDF which does not adapt well to the casual expressions and arbitrary writing common in microblogs, especially when performing a content search for a specific topic. The Aho–Corasick method is a string-matching algorithm that can be used to perform content searches. The experimental results show that the traditional search methods based on IF-IDF or string matching perform worse than do the learning-based methods. DSSM and CLSM learn semantic features based on deep neural networks to search target content by the similarities calculated from latent semantic spaces. However, these two deep learning search methods model the similarities between queries and contents based on global semantic feature representations, which are not robust to the semantic noise produced by the casual expressions and arbitrary writing in microblogs. RankNet and ListNet are two learning-to-rank approaches based on pairwise and listwise learning, respectively. They are also used by search engines in practice. RankNet is approximated by a classification problem that relies on labeled training data. ListNet is a listwise learning-to-rank approach that tries to directly optimize the values of evaluation measures. However, it is difficult for this approach to perform such optimizations without approximations or bounds because most of the evaluation measures are not continuous functions. Furthermore, another information search method (MDPRank) based on Markov decision process is tested as baseline method too. As shown in Tables 3 and 4, the performances of MDPRank on NDCG and MAP are close to MDPMS. However, MDPRank adopts the policy gradient algorithm of an on-policy strategy in reinforcement learning. The method forms microblog contents as actions to be chosen makes the policy space more complicated in gradient calculations than the off-policy methods for information search in social networks.

We also conducted the search experiment on another query set with more query items (2000). The NDCG and MAP evaluations are computed following the definitions in Eq. (5),

Table 4 Comparison of MDPMS and baseline methods on average MAP@n

	MAP			
	@5	@10	@15	@20
BM25	0.534	0.608	0.626	0.618
Aho–Corasick	0.453	0.423	0.411	0.437
RankNet	0.610	0.627	0.648	0.650
ListNet	0.619	0.632	0.690	0.706
DSSM	0.635	0.648	0.648	0.635
CLSM	0.612	0.649	0.673	0.672
MDPRank	0.622	0.641	0.706	0.703
MDPMS	0.640	0.654	0.720	0.738

Eq. (12), and Eq. (13). The average NDCG and MAP scores for the top 5, top 10, top 15 and top 20 results are presented in Tables 5 and 6.

As shown in Tables 5 and 6, the MDPMS NDCG evaluation values of 2000 queries are lower than the ones on 100 queries by 0.02 to 0.07, while its MAP evaluation values are reduced by 0.02 to 0.19. For the baseline methods, the evaluation values on NDCG and MAP are reduced to varying degrees. However, the numerical distributions of Tables 5 and 6 are consistent compared with those of Tables 3 and 4. As shown in Tables 5 and 6, RankNet performs better than the other selected methods for Top 15 in NDCG and Top 15 and 20 in MAP. We decide to select every 5 documents of contents in the ranking list as a step. RankNet shows better MAP performances at Top 15 and Top 20, of which the situation illustrates RankNet returned more accurate results according to the query. Furthermore, these results are ranked at the front positon. However, according to the NDCG evaluations, the proposed method returns more related contents overall.

The key factors of microblog search on specific topics contain effective strategies for matching, ranking and analyzing the semantic features of microblog contents. In contrast to traditional web search problems, searching microblog content for specific topics requires intelligent search strategies and different content features based on user-perceived search utility. Another difference involves the semantic features analysis processes to match content appropriately based on the queries.

The proposed method defines a microblog search as an MDP. Under this definition, the microblog search is constructed as a process of choosing and ranking results dynamically. NDCG is applied to define the reward to that forms the reinforcement during the training phase. The reward process is modeled as a part of the MDP state. As the definition of the reward, measures are evaluated dynamically based on action-dependent rewards for different microblog contents. Content analysis of semantic features on specific topics is an indispensable component for searching and ranking content. Because of the casual expression and arbitrary writing characteristics in microblog content, a deep Q network is designed to analyze semantic features implied by words representative of a specific topic based on deep Q learning.

We take the top 5 results on the topic of "Tianjin explosions" from microblogs searched by the proposed method MDPMS as an example. The query and these top 5 results are presented in Table 7.

As shown in Table 7, the research results can reflect the semantics expressed by the query content. The expreriments results show the effectiveness of MDPMS. Some other aspects are

Table 5 Comparison of MDPMS and baseline methods on average NDCG@n of 2000 queries

	NDCG			
	@5	@10	@15	@20
BM25	0.450	0.499	0.638	0.639
Aho–Corasick	0.508	0.698	0.515	0.546
RankNet	0.640	0.569	0.719	0.664
ListNet	0.592	0.612	0.534	0.507
DSSM	0.553	0.494	0.620	0.654
CLSM	0.601	0.681	0.666	0.661
MDPRank	0.541	0.616	0.695	0.697
MDPMS	0.688	0.703	0.674	0.843

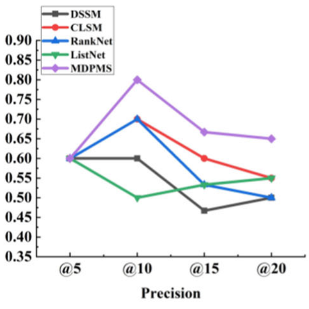
Table 6 Comparison of MDPMS and baseline methods on average MAP@n of 2000 queries

	MAP			
	@5	@10	@15	@20
BM25	0.584	0.588	0.595	0.608
Aho–Corasick	0.301	0.464	0.433	0.433
RankNet	0.612	0.625	0.601	0.661
ListNet	0.420	0.619	0.465	0.689
DSSM	0.514	0.425	0.470	0.520
CLSM	0.485	0.535	0.653	0.526
MDPRank	0.553	0.543	0.557	0.532
MDPMS	0.615	0.604	0.545	0.546

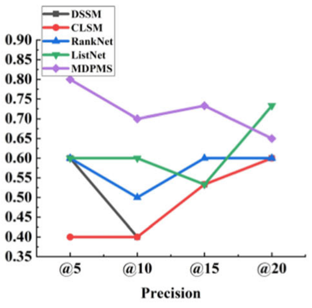
also worth discussing. In microblog contents posted by users, some subtopics of the main topic inevitably exist; the second and the fifth results show this the phenomenon. The subtopics include a series of discussions caused by the main topic. However, the results are similar to the query topic; therefore, they are found by the content searching for the security topic in microblog entries.

Table 7 The top 5 search results on the topic of "Tianjin explosions"

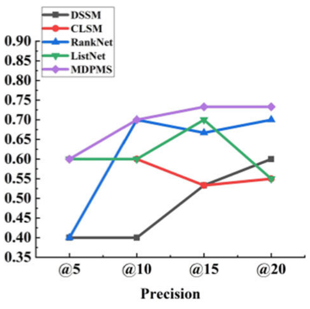
The query	#天津仓库爆炸事故#【 圆数公里有强烈震感】12日23时30分左右, 津滨海新区开发区周边瑞海危险品仓库发生爆炸。 据初步统计, 故已造成14人死亡,400余人受伤。 爆炸喷发火球, 时引发周边企业二次爆炸, 圆数公里有强烈震感。 (#Tianjin warehouse explosion accident#[A strong earthquake occurred within in a few kilometers of the warehouse] At approximately 23:30 on the 12th, the dangerous goods warehouse in Ruihai around the Tianjin Binhai New Area Development Zone exploded. According to preliminary reports, the accident caused 14 deaths and resulted in more than 400 injuries. Fireballs erupted from the explosion, and caused secondary explosions in surrounding enterprises; in addition, a strong earthquake occurred within a few kilometers.)
The results	Top 1: 天津港危险品仓库特别重大火灾爆炸事故死亡人数已上升至104人 (The number of deaths from the accidental explosion and fire in the Tianjin Port dangerous goods warehouse has risen to 104) Top 2: #天津塘沽大爆炸#转自一位医生朋友的朋友圈, 来对天津以及此事件受伤的人们很关心和同情, 在什么都不想说了…… (#Tianjin Tanggu explosion# Based on information from a doctor friend, I am very concerned and sympathetic to Tianjin and the people injured in this incident. But I can't reveal anything...) Top 3: 天津爆炸中6名牺牲消防员照片公布4人为90后(图) Photograph of 6 firefighters sacrificed in the Tianjin Explosion 4 of whom were born After 1990 (Photo) Top 4: #天津塘沽大爆炸#今年年初才去了天津, 津是一个很美丽的城市, 望伤亡人数不要再增加了, 消防员致敬。 (#Tianjin Tanggu explosion# I visited Tianjin at the beginning of this year. It is a very beautiful city. I hope that the number of casualties will not increase further, and I salute the firefighters.) Top 5: #天津塘沽大爆炸#愿伤亡人数不再上升!愿逝者安息~为伤者祈福消防官兵一定要平安!!为冲在一线的英雄们祈祷!太伟大了... (May the number of casualties rise no higher! May the dead rest in peace. Pray for the wounded and that the firefighters and soldiers stay safe! Pray for the heroes in the line! It's too much…)



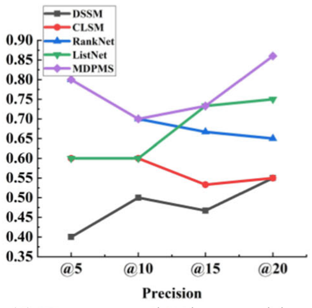
(a) The cross-validation precision on the first validation



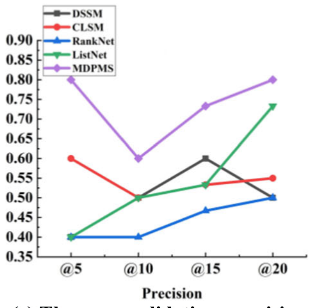
(b) The cross-validation precision on the second validation



(c) The cross-validation precision on the third validation



(d) The cross-validation precision on the fourth validation



(e) The cross-validation precision on the fifth validation

Figure 6 The precision of the 5-fold cross-validation

5.5 Cross-validation

We conducted a *k*-fold cross-validation [27] experiment to verify the effectiveness of the proposed method from a machine learning aspect. We adopted $k = 5$ to randomly partition the original dataset into 5 subdatasets of equal size. There is no overlap between these 5 subdatasets. In accordance with *k*-fold cross-validation, there are 5 validations (each subdataset is retained as a validation subdataset once) and the other 4 subdatasets are used as the training set. The prediction performances resulting from this cross-validation are presented in Figure 6, and the average precision values are shown in Table 8. As comparison methods for this cross-validation, we selected the existing learning-based methods, including DSSM, CLSM, RankNet and ListNet, as mentioned above.

Table 8 The average precision values from the 5-fold cross-validation

	Precision			
	@5	@10	@15	@20
DSSM	0.480	0.480	0.520	0.550
CLSM	0.560	0.560	0.546	0.560
RankNet	0.560	0.600	0.587	0.590
ListNet	0.560	0.560	0.606	0.663
MDPRank	0.689	0.689	0.683	0.709
MDPMS	0.720	0.700	0.720	0.739

As shown in Figure 6 and Table 8, the proposed MDPMS method outperforms the other selected learning-based methods. RankNet and ListNet focus on ranking from the pairwise and listwise aspects, respectively to solve the search problem. These two methods performed better when searching for related security topic content in microblogs during the second, third and fourth validations. DSSM and CLSM solve the problem from the aspect of semantic matching based on maximizing the click probability of the relevant documents. These methods are unable to meet the requirements of microblog searching because of the different semantic features of microblogs compared to those of web search. The proposed method MDPMS adopts the Markov decision process of reinforcement learning to search security-topic-related content for an appropriate user-perceived utility. Its performances during the cross-validation show its effectiveness for security-related content search tasks in microblogs.

6 Conclusions

In this paper, we proposed a method based on reinforcement learning (MDPMS) intended for searching for specific topics in microblogs. The method models the microblog search for specific topics as an MDP. A novel deep Q network combined with CNN and LSTM was designed to analyze the local semantic features for the target topics which were used to select appropriate actions (choose or skip) for the microblog entries to assemble the search results. The method evaluates content relevance dynamically through reinforcement learning instead of ranking based solely on similarities. Following the MDP reinforcement learning process, a reward based on NDCG was defined to model user-perceived search utility. In contrast to traditional web search methods, the proposed method focuses on intelligent strategies for searching and evaluating content relevance in accordance with typical microblog data features. The results of experiments based on real-world data showed that the proposed method outperformed the other baseline methods. The results also verified that intelligent search strategies and evaluations of content relevance are important to perform microblog searches on specific topics.

Acknowledgments This work was supported by the National Natural Science Foundation of China (NSFC) under Grant (No.61772083, No.61532006, No. 61877006, No. 61802028), in part by the Fundamental Research Funds for the Central University (No.2018RC44), in part by the Director Foundation of Beijing Key Laboratory of Intelligent Telecommunication Software and Multimedia (No.ITSM20180102).

References

1. Agarwal, M.K., Bansal, D., Garg, M., et al.: Keyword search on microblog data streams: finding contextual messages in real time[C]. In: Proceedings of 19th International Conference on Extending Database Technology (EDBT), pp. 15–18 (2016)
2. Asadi, N., Lin, J.: Fast candidate generation for real-time tweet search with bloom filter chains[J]. ACM Transactions on Information Systems (TOIS). **31**(3), 13 (2013)
3. Basu, M., Roy, A., Ghosh, K., et al.: A novel word embedding based stemming approach for microblog retrieval during disasters[C]. In: European Conference on Information Retrieval, pp. 589–597. Springer, Cham (2017)
4. Basu, M., Roy, A., Ghosh, K., et al.: Microblog retrieval in a disaster situation: a new test collection for evaluation[C]. SMERP@ ECIR. 22–31 (2017)
5. Borisov, A., Markov, I., de Rijke, M., et al.: A neural click model for web search[C]. In: Proceedings of the 25th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, pp. 531–541 (2016)

6. Burges, C., Shaked, T., Renshaw, E., et al.: Learning to rank using gradient descent[C]. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 89–96. ACM (2005)
7. Busch, M., Gade, K., Larson, B., et al.: Earlybird: real-time search at twitter[C]//data engineering (ICDE), 2012 IEEE 28th international conference on. IEEE. 1360–1369 (2012)
8. Calderone, D., Sastry, S.S.: Markov decision process routing games[C]//Proceedings of the 8th International Conference on Cyber-Physical Systems. ACM. 273–279 (2017)
9. Cao, Z., Qin, T., Liu, T.Y., et al.: Learning to rank: from pairwise approach to listwise approach[C]. In: Proceedings of the 24th International Conference on Machine Learning, pp. 129–136. ACM (2007)
10. Chen, C.C., Wang, S.D.: An efficient multicharacter transition string-matching engine based on the aho-corasick algorithm[J]. ACM Transactions on Architecture and Code Optimization (TACO). **10**(4), 25 (2013)
11. Chen, C., Li, F., Ooi, B.C., et al.: Ti: an efficient indexing mechanism for real-time search on tweets[C]. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, pp. 649–660. ACM (2011)
12. Chen, Q., Hu, Q., Huang, J., et al.: TAKer: fine-grained time-aware microblog search with kernel density estimation[J]. IEEE Trans. Knowl. Data Eng. **30**(8), 1602–1615 (2018)
13. De Maio, C., Fenza, G., Gallo, M., et al.: Time-aware adaptive tweets ranking through deep learning[J]. Futur. Gener. Comput. Syst. (2017)
14. Dolotta, T.A.: Data Processing in 1980–1985[M]. Wiley (1976)
15. Dzida, W., Herda, S., Itzfeldt, W.D.: User-perceived quality of interactive systems[J]. IEEE Trans. Softw. Eng. **SE-4**(4), 270–276 (1978)
16. Feng, S., Song, K., Wang, D., et al.: A word-emoticon mutual reinforcement ranking model for building sentiment lexicon from massive collection of microblogs[J]. World Wide Web Internet and Web Information Systems. **18**(4), 949–967 (2015)
17. Feng, S., Wang, Y., Liu, L., et al.: Attention based hierarchical LSTM network for context-aware microblog sentiment classification[J]. World Wide Web Internet and Web Information Systems. **2018**, 1–23
18. Graves A. Generating Sequences with Recurrent Neural Networks[J]. arXiv preprint arXiv:1308.0850, 2013
19. Guo, J., Fan, Y., Ai, Q., et al.: A deep relevance matching model for ad-hoc retrieval[C]. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 55–64. ACM (2016)
20. Hasanain, M., Elsayed, T.: Query performance prediction for microblog search[J]. Inf. Process. Manag. **53**(6), 1320–1341 (2017)
21. Herranz, L., Jiang, S., Li, X.: Scene recognition with CNNs: objects, scales and dataset bias[C]. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 571–579 (2016)
22. Hochreiter, S., Schmidhuber, J.: Long short-term memory[J]. Neural Comput. **9**(8), 1735–1780 (1997)
23. Huang, P.S., He, X., Gao, J., et al.: Learning deep structured semantic models for web search using clickthrough data[C]. In: Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management, pp. 2333–2338. ACM (2013)
24. Huang, J., Peng, M., Wang, H., et al.: A probabilistic method for emerging topic tracking in microblog stream[J]. World Wide Web Internet and Web Information Systems. **20**(2), 325–350 (2017)
25. Keyhanipour, A.H., Moshiri, B., Rahgozar, M., Oroumchian, F., Ansari, A.A.: Integration of data fusion and reinforcement learning techniques for the rank-aggregation problem[J]. Int. J. Mach. Learn. Cybern. **7**(6), 1131–1145 (2016)
26. Keyhanipour, A.H., Keyhanipour, A.H., Moshiri, B., et al.: Learning to rank with click-through features in a reinforcement learning framework[J]. International Journal of Web Information Systems. **12**(4), 448–476 (2016)
27. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection[C]// Proceedings of the International Joint Conferences on Artificial Intelligence. **14**(2), 1137–1145 (1995)
28. Kou, F., Du, J., He, Y., Ye, L.: Social network search based on semantic analysis and learning[J]. CAAI Transactions on Intelligence Technology. **1**(4), 293–302 (2016)
29. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning[J]. Nature. **521**(7553), 436–444 (2015)
30. Liu, X., Gao, J., He, X., et al.: Representation learning using multi-task deep neural networks for semantic classification and information retrieval[C]. HLT-NAACL. 912–921 (2015)
31. Luo, J., Zhang, S., Yang, H.: Win-win search: dual-agent stochastic game in session search[C]. In: Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 587–596. ACM (2014)
32. Mao, J., Liu, Y., Luan, H., et al.: Understanding and predicting usefulness judgment in web search[C]. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1169–1172. ACM (2017)
33. Mikolov, T., Chen, K., Corrado, G., et al.: Efficient Estimation of Word Representations in Vector Space[J]. arXiv preprint arXiv:1301, vol. 3781, (2013)
34. Miranda, F., Lins, L., Klosowski, J.T., Silva, C.T.: TOPKUBE: a rank-aware data cube for real-time exploration of spatiotemporal data[J]. IEEE Trans. Vis. Comput. Graph. **24**(3), 1394–1407 (2018)

35. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning[J]. Nature. **518**(7540), 529–533 (2015)
36. Mnih, V., Badia, A.P., Mirza, M., et al.: Asynchronous methods for deep reinforcement learning[C]. In: International Conference on Machine Learning, pp. 1928–1937 (2016)
37. Nguyen, D.T., Jung, J.E.: Real-time event detection for online behavioral analysis of big social data[J]. Futur. Gener. Comput. Syst. **66**, 137–145 (2017)
38. Olteanu, A., Castillo, C., Diaz, F., et al.: CrisisLex: a lexicon for collecting and filtering microblogged communications in crises[C]. In: Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media, pp. 376–386 (2014)
39. Puterman, M.L.: Markov decision processes[J]. Handbooks in Operations Research and Management Science. **2**, 331–434 (1990)
40. Robertson, S., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond[J]. Foundations and Trends® in Information Retrieval. **3**(4), 333–389 (2009)
41. Rodriguez Perez, J.A.: Microblog Retrieval Challenges and Opportunities[D]. University of Glasgow (2018)
42. Schütze, H.: Introduction to information retrieval[C]. Proceedings of the International Communication of Association for Computing Machinery Conference. (2008)
43. Severyn, A., Moschitti, A.: Learning to rank short text pairs with convolutional deep neural networks[C]. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 373–382. ACM (2015)
44. Shen, Y., He, X., Gao, J., et al.: A latent semantic model with convolutional-pooling structure for information retrieval[C]. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pp. 101–110. ACM (2014)
45. Shen, Y., He, X., Gao, J., et al.: Learning semantic representations using convolutional neural networks for web search[C]. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 373–374. ACM (2014)
46. 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., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D.: Mastering the game of go with deep neural networks and tree search[J]. Nature. **529**(7587), 484–489 (2016)
47. Singla, R., Modha, S., Majumder, P., et al.: Information extraction from microblog for disaster related event[C]//SMERP@ ECIR. 85–92 (2017)
48. Song, X., Jiang, S., Herranz, L.: Multi-scale multi-feature context modeling for scene recognition in the semantic manifold[J]. IEEE Trans. Image Process. **26**(6), 2721–2735 (2017)
49. Song Z, Zhang L, Liu T, et al. Ranking learning algorithm of information retrieval based on WeChat public numbers[C]//Proceedings of the 6th International Conference on Information Engineering. ACM, 2017: 4
50. Sutton, R.S., Barto, A.G.: Reinforcement learning: an introduction (2nd ed)[M]. Cambridge. MIT press. (2016)
51. Wang, S., Huang, S., Liu, T.Y., et al.: Ranking-oriented collaborative filtering: a listwise approach. [J]. ACM Transactions on Information Systems (TOIS). **35**(2), 10 (2016)
52. Wang, Y., Huang, H., Feng, C.: Query expansion based on a feedback concept model for microblog retrieval[C]. In: Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, pp. 559–568 (2017)
53. Wei, Q., Lewis, F.L., Sun, Q., Yan, P., Song, R.: Discrete-time deterministic Q-learning: a novel convergence analysis[J]. IEEE Transactions on Cybernetics. **47**(5), 1224–1237 (2017)
54. Wei, Z., Xu, J., Lan, Y., et al.: Reinforcement learning to rank with Markov decision process[C]. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 945–948. ACM (2017)
55. Xia, L., Xu, J., Lan, Y., et al.: Adapting markov decision process for search result diversification[C]. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 535–544. ACM (2017)
56. Xia, F., Yu, C., Xu, L., et al.: Top-k temporal keyword search over social media data[J]. World Wide Web Internet and Web Information Systems. **20**(5), 1049–1069 (2017)
57. Xingjian, S.H.I., Chen, Z., Wang, H., et al.: Convolutional LSTM network: a machine learning approach for precipitation now casting[C]. Adv. Neural Inf. Proces. Syst. 802–810 (2015)
58. Xu, J., Xia, L., Lan, Y., et al.: Directly optimize diversity evaluation measures: a new approach to search result diversification[J]. ACM Transactions on Intelligent Systems and Technology (TIST). **8**(3), 41 (2017)

59. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval[C]//ACM SIGIR Forum. ACM. **51**(2), 268–276 (2017)
60. Zhang, X., He, B., Luo, T., et al.: Query-biased learning to rank for real-time twitter search[C]. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, pp. 1915–1919. ACM (2012)
61. Zhang, D., Nie, L., Luan, H., et al.: Compact indexing and judicious searching for billion-scale microblog retrieval[J]. ACM Transactions on Information Systems (TOIS). **35**(3), 27 (2017)
62. Zhang, R., Jin, Z., Liu, X.: A study on the analysis model of the ranking of the theme of Weibo[J]. Int. J. Pattern Recognit. Artif. Intell. **32**(03), 1851003 (2018)
63. Zheng, N., Jin, M., Hong, H., Huang, L., Gu, Z., Li, H.: Real-time and precise insect flight control system based on virtual reality[J]. Electron. Lett. **53**(6), 387–389 (2017)

Affiliations

Nan Zhou[1] • **Junping Du**[1] • **Xu Yao**[1] • **Wanqiu Cui**[1] • **Zhe Xue**[1] • **Meiyu Liang**[1]

[1] Beijing Key Lab of Intelligent Telecommunication Software and Multimedia, School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China