Electronic Preprint for Journal of Information Processing Vol.28

Regular Paper

Searching for Microblogs Referring to Events by Deep Dynamic Query Strategies

Jun-Li Lu[1,a)] Makoto P. Kato[2,3,b)] Takehiro Yamamoto[4,c)] Katsumi Tanaka[1,d)]

Received: September 8, 2019, Accepted: January 5, 2020

Abstract: We address the problem of searching for microblogs referring to events, which are difficult to find because microblogs may refer to events without using event's contents and a searcher may not use suitable queries for a search engine. We therefore propose a dynamic search process based on MDP that takes query strategies optimized for the current search state. As key components of the dynamic search process, we propose an RNN-based model for predicting long-term returns of a search process, and a DNN-based model that tries to match between the representations of microblogs and those of events for identifying relevant microblogs. Experimental results suggest that the dynamic search process could effectively search for microblogs, especially for implicitly referred events. Moreover, we show high applicability of our proposed approach to unseen events for which any relevant microblogs were not available in the training phase.

Keywords: microblog search, reinforcement learning, RNN, DNN

1. Introduction

Searching for microblogs referring to a particular event is useful when a user intends to survey people's opinions and feelings that are not reported in news articles. Most studies of event-focused microblog searching tend to heavily rely on the event's contents, which are often keywords or named entities of events. However, this approach is unsuitable when the reference to the event is *implicit* and excludes the event's contents. We discuss the challenging points of the microblog search problem as follows. (1) Searching for microblogs implicitly or explicitly referring to events. We explain the difficulty of implicit references as follows. For example, a user may describe the result of a New York Yankees vs. Boston Red Sox game as "A really disappointing game." In this case, search keywords such as "yankees red sox" would not find such microblogs, while more general keywords such as "game" would produce too many search results. (2) The limited search results provided by the search engine for microblogs. A search engine usually has its own ranking criteria for search results or has the limited number of return. For example, even if one makes effective queries for text contents but he/she cannot get the relevant microblogs posted in the past or specific regions because the search engine prefers the latest or global ones in the top of search results. Furthermore, as the fraction of microblogs referring to a particular event can be extraordinarily small in many cases, finding effective queries under the API call limit imposed by the microblog hosting company is a challenging task for both humans and algorithms. (3) Last but not least, we maximize *search efficiency*, which is measured by the recall of relevant microblogs for a fixed amount of issued queries.

To this end, we propose a dynamic search process that takes query strategies optimized for the current search state. This approach monitors the current search state and takes the most effective query strategy in the next step. This approach consists of the following three components. (a) We propose exploratory and exploitative query strategies to search for microblogs implicitly or explicitly referring to events. More concretely, the query strategy can indicate whether queries for relevant microblogs with high novelty (e.g., by adding novel words of recent search results to the current query) should be explored or whether queries known to be effective (e.g., by using frequently occurring words in the previous search results) should be exploited. (b) To accurate monitor the search process, we propose an RNN-based model that predicts long-term returns of a search process based on not only current search states but also the history of search states and query strategies. Subsequently, we train the dynamic search process model using deep Q-learning [1] (a kind of reinforcement learning) that maximizes the expected return of every issued query. (c) To increase the applicability of this research, i.e., the search process can be applied to not only events seen during training phase but also to unseen or new events, we optimize the query strategies for events using the estimated relevance of microblogs by a DNN-based model. The proposed DNN model is based on transfer learning [2] and tries to match between the

[1] Graduate School of Informatics, Kyoto University, Kyoto 606–8501, Japan
[2] Faculty of Library, Information and Media Science, University of Tsukuba, Tsukuba, Ibaraki 305–8550, Japan
[3] JST, PRESTO, Chiyoda, Tokyo 102–0076, Japan
[4] School of Social Information Science, University of Hyogo, Kobe, Hyogo 651–2197, Japan
[a)] junli.lulu@gmail.com
[b)] mpkato@slis.tsukuba.ac.jp
[c)] t.yamamoto@sis.u-hyogo.ac.jp
[d)] tanaka.katsumi.85e@st.kyoto-u.ac.jp

© 2020 Information Processing Society of Japan

representations of microblogs and seen/unseen events using nonlinear dual-transformations.

The motivation of applying reinforcement learning in this research is explained as follows. We explain by the components of agent, environment, action, state, and return that are used in a reinforcement learning algorithm [3]. For the microblog search problem, the given search engine (environment) can be unknown to the proposed search process (agent). For example, the ranking criterion of search results from the engine is unknown to the search process. Meanwhile, as the search results are usually limited and the fraction of relevant microblogs can be extraordinarily small, searching for relevant microblogs on the engine (exploring the environment) can be much difficult. Therefore, we apply reinforcement learning to assist the search process to take effective query strategies (actions) based on the observed search states (states) and the predicted long-term returns (returns).

We conducted simulation-based experiments using 28,406 tweets randomly sampled from Twitter. The experimental results suggested that (1) our RNN-based model effectively estimated the long-term returns of a search process, (2) our DNN-based model effectively identified relevant microblogs, and (3) the dynamic query strategies were more effective than stationary strategies, and outperformed corpus-based and blog-to-event-based search methods, especially when events were implicitly referred to.

The contributions of this work are summarized as follows:

- We proposed a dynamic search process based on the Markov Decision Process (MDP) that issues different queries depending on the current search state, and an RNN-based model that estimates the long-term returns of a search process.
- We proposed a DNN-based model that estimates the relevance of the microblogs through nonlinear dual-transformations.
- We demonstrated the effectiveness of the main components of the dynamic search process, and showed that dynamic query strategies are more effective than stationary strategies and baseline search methods, especially for implicit references.

The remainder of this paper is organized as follows. Related works are discussed in Section 2, and the microblog search problem is presented in Section 3. Our dynamic search process based on MDP and its key building blocks are introduced in Section 4. Section 5 discusses the results of the experiments using real-world microblog data, evaluates the proposed method, and presents a case study. The paper concludes with Section 6.

2. Related Work

Crawling for Event Information

Crawling of event information from multiple data sources, such as documents, web pages, and microblogs, has been extensively studied. These studies [4], [5], [6], [7], [8] utilized the features of multiple attributes of data sources, such as keywords or named entities of textual contents, corresponding spatial locations, temporal information, and the information of writers. Crawling information additionally includes the relationships and correspondences between the target events and data sources. In some studies [8], the credibility or accuracy of the extracted event's information was improved by evaluating the data sources and the metadata (e.g., web media platforms of the data sources or the initial seeds of data), and mining typical patterns of the data. Other studies [9], [10] provide the event information in real time by extracting up-to-date information from social media platforms. For example, the proposed methods efficiently identify the typical patterns or unusual bursty streams in multiple contexts (text, location, or time) of Twitter microblogs. Our proposed method is applicable not only to the general crawling problem, but also to data sources (e.g., microblogs) that implicitly refer to events without including their contents. We also target for unseen events, which are especially challenging because their relevant microblogs have not been seen by a searcher.

Crawler Based on MDP with Reinforcement Learning

The relevant contents of a target object (e.g., an event) can be effectively crawled by focused crawlers that utilize MDP with reinforcement learning [3], [11], [12], [13], [14]. In popular focused crawlers, the states of the MDP are the extracted results such as web pages or database records, and the actions are simple movements, such as following the links of web pages or issuing queries from records. However, we argue that these simple states and actions might be inappropriate for search processes requiring search efficiency or the finding of implicitly referred results.

We therefore propose a dynamic search process that efficiently finds the implicitly referred results through the proposed search states and query strategies. Therefore, the search process can be guided to explore results unseen in the previous results or to exploit deeply for results similar to the found ones recently. We also propose an RNN-based model that accurately estimates the long-term returns in a sequence of consecutive time steps, whereas common models estimate the returns in a single time step. Moreover, we can optimize the actions of an online search process by estimating the immediate rewards of the actions, whereas a common search process must rely only on the offline trained policy. Last but not least, we apply state-of-the-art reinforcement learning approaches [1], [15], [16] (deep Q-learning, double Q-learning, and experience replay) to ensure the reliability of the estimation results.

Zero-shot Learning

To identify the microblogs with relevance to the target events, especially those of the unseen events, we utilize a transfer-learning technique called zero-shot learning (ZSL) [17]. After such transfer learning [18], a classifier can categorize instances into a class for which no instances exist in the training data. ZSL has been applied to the learning of intermediate attribute classifiers [19], [20], [21], [22], learning a mixture of seen class proportions [23], [24], [25], and learning of the compatibility among heterogeneous spaces [26], [27], [28]. A typical ZSL framework [28] links the unseen images to the correct text label by mapping the image representations in convolutional neural network (CNN) models to the text representations in Word2vec models. We therefore propose a DNN-based model and train it to identify relevant microblogs by the ZSL strategy. The DNN model applies nonlinear dual-transformations that increases the accuracy of mapping between the microblog and event represen-

© 2020 Information Processing Society of Japan

tations, which are extracted from heterogeneous spaces.

3. Microblog Search Problem

We first introduce several terms that are important for defining the target problem.

Microblog A microblog is posted by a user of a social media platform, such as a tweet posted by a Twitter user. A microblog often includes textual content, a post time, and a post location.

Event A common definition of an event is "a thing that happens, for a specific time and place, and can be observed by human" [29], [30], such as news, holidays, disasters. In this research, we study on news, a common type of events that are referred to by microblogs.

Reference A microblog b refers to an event e if the writer posted b to reply to e. Note that evidence of a reply can be a hyperlink or a reply-link from b to e. A reference can be either *explicit* or *implicit*. For example, a microblog "Congrats to Red Sox, the winner of the 2018 World Series Championship!" *explicitly* refers to the event "The Red Sox beat the Dodgers in the 2018 World Series", while another microblog "Many congrats to Bosox." *implicitly* refers to the same event.

Finally, the present paper tackles the following problem:

Definition 3.1 (Searching for microblogs referring to events)**.** Given an event e and a search engine for microblogs, the problem is to retrieve as many microblogs referring to e as possible by issuing queries to the search engine, where the number of search results for one-time query is k and k is extremely small compared to the number of all microblogs. The success of an algorithm in solving this problem is measured by the recall of relevant microblogs after issuing n queries.

4. Dynamic Search Process

4.1 MDP-based Search Process

To efficiently search for microblogs referring to an given event, we propose a dynamic search process based on MDP [3].

An MDP is composed of states S, actions A, a transition function T, a reward function R, and a policy π. Let $s_t \in S$ be a state at time t. The process takes an action $a_t \in A$ from a probability distribution $\pi(a_t|s_t)$, and moves to the next state s_{t+1} as determined by the transition function $T(s_t, a_t)$, i.e., $s_{t+1} = T(s_t, a_t)$. The process receives a reward $R(s_t, a_t)$ by taking the action a_t at the state s_t. The goal of the MDP is to find the policy that maximizes the cumulative rewards.

The query strategies of a search process often include (but are not limited to) *exploration*, which devises queries for retrieving microblogs not retrieved in past queries, and *exploitation*, which continues to retrieve microblogs similar to those found by recent queries. The dynamics of the search process highly depend on both the given event and the current search state, such as the numbers of relevant and newly found microblogs in the latest search results. These characteristics of the search process can be well modeled by MDP. At each time step, the search process chooses a query strategy (or an action) by following the optimized policy. The chosen query strategy then generates a query to retrieve new search results. The retrieved results update the state and issue a reward based on the number of relevant results.

Below, we explain how each component of an MDP can be instantiated in the dynamic search process.

States

The state $s_t = (\mathbf{v}_t, \mathbf{v}_t^+, d_t^{\text{rel}}, d_t^{\text{new}})$ of the search process at time t can be used to measure the degree of exploration and exploitation, i.e., whether the process is retrieving more unseen microblogs than familiar ones, or vice versa.

$\mathbf{v}_t$ represents the difference between the search results retrieved by a query at time t (denoted by q_t), and the previous search results, i.e., those retrieved by q_{t-1}. More precisely, let $\mathbf{b} = (b_1, b_2, \ldots, b_m)$ be a vector representing m features of a microblog, where b_1 and b_2 are the textual content and post time of a microblog, respectively. The i-th value of $\mathbf{v}_t$ is defined as:

$$v_{t,i} = \|\bar{b}_{t,i} - \bar{b}_{t-1,i}\|, \tag{1}$$

where $\bar{b}_{t,i}$ is the mean of the microblogs B_t for the i-th feature: $\bar{b}_{t,i} = \frac{1}{|B_t|}\sum_{\mathbf{b}\in B_t} b_i$, and B_t is a set of microblogs retrieved by the query at time t.

$\mathbf{v}_t^+$ represents the difference between the *relevant* search results at time t and the *relevant* search results at time $t-1$. Thus, the i-th value of $\mathbf{v}_t^+$ is defined as:

$$v_{t,i}^+ = \|\bar{b}_{t,i}^+ - \bar{b}_{t-1,i}^+\|, \tag{2}$$

where $\bar{b}_{t,i}^+ = \frac{1}{|B_t^+|}\sum_{\mathbf{b}\in B_t^+} b_i$, and B_t^+ is the set of *relevant* microblogs retrieved by q_t.

d_t^{rel} is the difference between the number of *relevant* search results at time t, and the number of *relevant* search results at time $t-1$: $d_t^{\text{rel}} = |B_t^+| - |B_{t-1}^+|$.

Finally, d_t^{new} is the difference between the number of newly found search results n_t at time t and the number of newly found search results at time $t-1$: $d_t^{\text{new}} = n_t - n_{t-1}$, where $n_t = \left|B_t - \bigcup_{t'=1}^{t-1} B_{t'}\right|$.

Note that $\mathbf{v}_t^+$ and d_t^{rel} are estimated except in the training phase when the relevant microblogs are known. When the ground truth is unavailable, the relevant microblogs can be estimated by our proposed DNN-based model as described in Section 4.4.

Actions

Our dynamic search process performs two types of actions, namely, exploitative and exploratory query strategies.

(1) **Exploitative Query Strategy:** This type of strategies uses a *salient* term or a *central* value in the latest search results as a query. For example, a query strategy uses the content term with the maximum term frequency-inverse document frequency (TF-IDF) score, or a post time close to the average post time in the latest search results as a query.

(2) **Exploratory Query Strategy:** This type of strategies uses a *serendipitous* term or an *outer* value in the latest search results as a query. For example, a query strategy uses the content term with a low TF-IDF score, or a post time far from the average post time in the latest search results as a query.

We can define both types of query strategies for each feature of a microblog. Therefore, the number of actions A is $2m$ (recall that m is the number of features). The details of the strategies depend on the implementation, and will be explained in Section 5.

© 2020 Information Processing Society of Japan

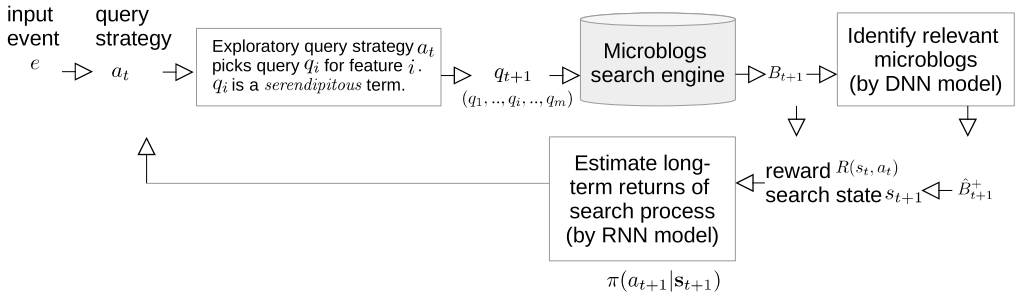


Fig. 1 Our dynamic search process searches for microblogs referring to target event e by issuing queries to the search engine. We select a query strategy a_t by policy $\pi(a_t|\mathbf{s}_t)$ and generate the query q_{t+1} for the next time, and retrieve microblogs B_{t+1}, identify the relevant microblogs $\hat{B}^+_{t+1}$, and transit to the next search state s_{t+1}.

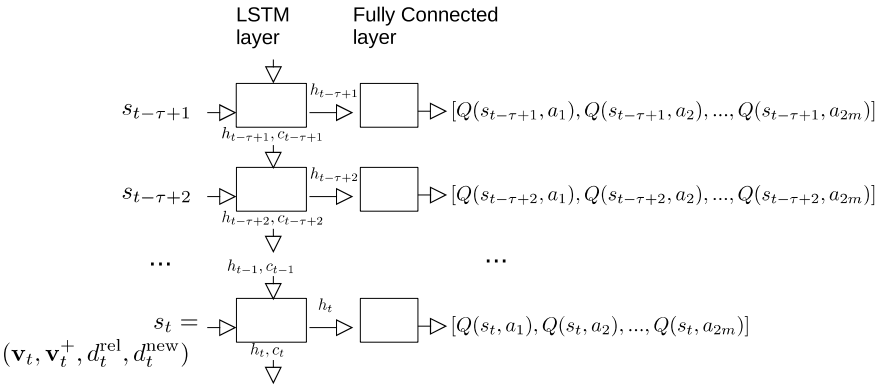


Fig. 2 Our RNN-based model predicts the long-term returns $Q(\mathbf{s}_t, a)$ of each query strategy $a \in A$ by using a sequence of search states $\mathbf{s}_t = (s_{t-\tau+1}, \ldots, s_t)$.

Transition

In the dynamic search process, a transition refers to a change from search state s_t to search state s_{t+1}, and is determined by the search results B_t, B^+_t, B_{t+1}, and B^+_{t+1}.

Reward

A reward is an immediate return of executing an action. In our method, the reward is the number of relevant microblogs in the search results; that is, $R(s_t, a_t) = |B^+_{t+1}|$. As the relevant microblogs are not known *a priori*, the relevance of the retrieved microblogs is estimated by the DNN-based model proposed in Section 4.4.

Policy

Policy describes the behaviors of the dynamic search process. Given the current state s_t and event e, the policy $\pi(a_t|s_t)$ selects an action $a_t \in A$ based on the long-term return of the search process.

Our search process is demonstrated in **Fig. 1**. Given an event e, we select a query strategy a_t based on the policy $\pi(a_t|\mathbf{s}_t)$ that considers the long-term return of applying a_t according to a sequence of search states $\mathbf{s}_t$, and generate query q_{t+1} for the next time. We then retrieve microblogs B_{t+1} and estimate the reward $|\hat{B}^+_{t+1}|$, and transit to the next search state s_{t+1}.

4.2 Deep Q-learning with the RNN Model

As explained above, the goal of the MDP is to learn the policy that maximizes the cumulative rewards. In the Q-learning algorithm [1], [15], [16], the long-term return is estimated by the Q function. The Q function then determines the policy π as follows:

$$\pi(a_t|s_t) = \frac{\exp(Q(s_t, a_t))}{\sum_{a \in A} \exp(Q(s_t, a))}. \tag{3}$$

Intuitively, the policy prefers actions with high long-term rewards.

The Q function is learned so that the following loss function is minimized:

$$\mathbb{E}\left[R(s_t, a_t) + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right], \tag{4}$$

where γ controls the importance between the immediate and future returns. Under this loss function, the Q function should approximate the sum of the immediate reward of the action a_t at the state s_t, and the future reward of the maximum value of the Q function at the next state s_{t+1}.

In standard Q-learning, the long-term reward is estimated only from the current state and action, not by the past states and actions. To overcome this drawback, we propose an RNN-based model in which the Q function estimates the long-term return from the history of states and actions. The Q function with this model determines the policy $\pi(a_t|\mathbf{s}_t)$ by considering the past states $\mathbf{s}_t = (\ldots, s_{t-1}, s_t)$, and is denoted by $Q(\mathbf{s}_t, a)$. The RNN model comprises a long short-term memory (LSTM) [31] layer and a fully connected (FC) layer, which is detailed in Section 4.3.

The RNN model is illustrated in **Fig. 2**. A sequence of states $\mathbf{s}_t = (s_{t-\tau+1}, s_{t-\tau+2}, \ldots, s_t)$ of length τ is input to the LSTM layer of h-dimensional hidden states. Inputting the hidden state at time t to a FC layer with a $h \times |A|$ matrix, we obtain the long-term reward $Q(\mathbf{s}_t, a)$ at time t for all $a \in A$.

The Q function is trained by a deep Q-learning algorithm. For clarity, we denote by $Q(\mathbf{s}, a; \mathbf{W})$ the Q function parameterized by $\mathbf{W}$. We then update Q by the following iterative process:

© 2020 Information Processing Society of Japan

(1) In each iteration, choose a_t at each time t by an ϵ-greedy selection method. This method either randomly selects a_t with probability ϵ or $a_t = \arg\max_a Q(\mathbf{s}_t, a; \mathbf{W})$ with probability $1 - \epsilon$.
(2) Compute the corresponding reward $R(s_t, a_t)$ of executing action a_t, and move to the next state s_{t+1}.

A large ϵ encourages exploration of various actions, whereas a small value encourages the selection of the best action for the current Q function.

Here we adopt *experience replay* [15], a common technique in deep Q-learning that randomly chooses from the previously observed state-action sequences and updates the Q function by the sampled sequences. We additionally employ *double Q-learning* [1] for stable learning, in which one model selects the actions and another model determines the value of the Q function under the selected actions. More formally, the loss function is now defined as:

$$\mathbb{E}\left[R(s_t, a_t) + \gamma Q(\mathbf{s}_{t+1}, a^*_{t+1}; \mathbf{W}') - Q(\mathbf{s}_t, a_t; \mathbf{W})\right], \tag{5}$$

where $a^*_{t+1} = \arg\max_a Q(\mathbf{s}_{t+1}, a; \mathbf{W})$. Note that Eq. (5) involves two different models. The model with $\mathbf{W}$ selects the action a^*_{t+1}, whereas the model with $\mathbf{W}'$ computes the value of the Q function for a^*_{t+1}. The parameter set W' is updated to W after every constant number of iterations.

4.3 An RNN Model that Predicts Long-Term Returns

We propose an RNN-based model that is used to implement the Q function $Q(\mathbf{s}, a; \mathbf{W})$ that computes the long-term returns for each query strategy $a \in A$.

To this end, the RNN model comprises an LSTM layer and a fully connected (FC) layer. The LSTM layer [32], [33] can remember previous search states and keeps track of the dependencies among the search states, for arbitrary and long time-intervals. The FC layer then transforms the output of the LSTM, which is the information of the required current and past search states, into the long-term returns of each query strategy a.

In Fig. 2, given our search state s_t at time t, s_t is first input to the LSTM layer with the hidden state h_{t-1} and memory cell state c_{t-1} [32]:

$$\begin{aligned}
f_t &= \sigma(\mathbf{W}_f s_t + \mathbf{U}_f h_{t-1} + \boldsymbol{\delta}_f),\\
i_t &= \sigma(\mathbf{W}_i s_t + \mathbf{U}_i h_{t-1} + \boldsymbol{\delta}_i),\\
o_t &= \sigma(\mathbf{W}_o s_t + \mathbf{U}_o h_{t-1} + \boldsymbol{\delta}_o),\\
c_t &= f_t \otimes c_{t-1} + i_t \otimes \tanh(\mathbf{W}_c s_t + \mathbf{U}_c h_{t-1} + \boldsymbol{\delta}_c),\\
h_t &= o_t \otimes \tanh(c_t).
\end{aligned} \tag{6}$$

Activation functions are sigmoid function (σ) and hyperbolic tangent function (tanh). $\otimes$ denotes the operation of element-wise product. LSTM updates the state c_t of the memory cell, which keeps tracking the dependencies among previous search states, by controlling the extent i_t to which the new value (from search state s_t) should be used and the extent f_t to which the existing cell value c_{t-1} should be remained, in the memory cell. The output of new hidden state h_t is computed based on the extent o_t to which the state c_t of the memory cell should be used.

The computed h_t is then input to the FC layer to compute the long-term returns for each query strategy $a \in A$:

$$[Q(s_t, a_1) \dots Q(s_t, a_{|A|})] = \mathrm{ReLU}(\mathbf{W}_a h_t + \boldsymbol{\delta}_a)^T, \tag{7}$$

where $\mathbf{W}_a$ is a matrix of dimension $|A| \times h$ and $\boldsymbol{\delta}_a$ is biases of dimension $|A| \times 1$. The FC layer uses the activation function of rectified linear unit (ReLU) [34]:

$$\mathrm{ReLU}([x_1 \dots x_{|A|}]^T) = [\max(0, x_1) \dots \max(0, x_{|A|})]^T, \tag{8}$$

where $x_i \in \mathbb{R}$.

Therefore, for the Q function $Q(\mathbf{s}, a; \mathbf{W})$, $\mathbf{W}$ comprises the trainable parameters below. In LSTM layer, the parameters are the matrices $\mathbf{W}_f$, $\mathbf{W}_i$, $\mathbf{W}_o$, $\mathbf{W}_c \in \mathbb{R}^{h \times s}$, the matrices $\mathbf{U}_f$, $\mathbf{U}_i$, $\mathbf{U}_o$, $\mathbf{U}_c \in \mathbb{R}^{h \times h}$, and the biases vectors $\boldsymbol{\delta}_f$, $\boldsymbol{\delta}_i$, $\boldsymbol{\delta}_o$, $\boldsymbol{\delta}_c \in \mathbb{R}^{h \times 1}$, where s is the dimension of search state s_t and h is LSTM dimension. In FC layer, the parameters are $\mathbf{W}_a$ and $\boldsymbol{\delta}_a$.

4.4 A DNN Model that Identifies Relevant Microblogs for Events

In our dynamic search process, the reward is defined as the number of relevant search results, and the relevance is assumed to be known in the previous subsection. This subsection proposes a DNN model that estimates the relevance of microblogs, even when the relevant microblogs are not known *a priori*, for the search process used in production. Both microblogs and events are first represented by their own vectors. The vector of a microblog represents the textual content, writer's profile, post location, and post time, while the vector of an event represents the event's content. Their vectors are transformed separately with different transform functions, into a space of the same dimension, and their relevance can be estimated by the inner product of their transformed ones. This technique estimates the relevance of microblogs to not only seen events in the training data, but also unseen events that are not in the training data. I.e., we propose to solve so-called *zero-shot learning* problem [17].

Specifically, the relevance of an event e and a microblog b is estimated by:

$$\begin{aligned}
F(e, b) &= \phi_e(\mathbf{e})^T \phi_b(\mathbf{b})\\
&= \mathrm{ReLU}(\mathbf{W}_e \mathbf{e} + \boldsymbol{\delta}_e)^T \mathrm{ReLU}(\mathbf{W}_b \mathbf{b} + \boldsymbol{\delta}_b).
\end{aligned} \tag{9}$$

$\mathbf{e}$ and $\mathbf{b}$ are the vector representations of e and b, with dimensions $n_e \times 1$ and $n_b \times 1$, respectively, which are detailed in Section 5.2. The function ϕ_e is a nonlinear transformation comprising a FC layer with the activation function of ReLU [34], i.e., $\phi_e(\mathbf{e}) = \mathrm{ReLU}(\mathbf{W}_e \mathbf{e} + \boldsymbol{\delta}_e)$, where $\mathbf{W}_e$ is a trainable matrix of dimension $n_e \times n_e$ and $\boldsymbol{\delta}_e$ is trainable biases of dimension $n_e \times 1$. The function ϕ_b is defined in the same way but has a matrix $\mathbf{W}_b$ of dimension $n_e \times n_b$ and biases $\boldsymbol{\delta}_b$ of dimension $n_e \times 1$.

The functions ϕ_e and ϕ_b are trained by pairwise ranking loss [17]. When an event e is paired with a relevant microblogs b, the pairwise ranking loss is given by

$$\sum_{e' \in E_{\mathrm{tr}}} [\Delta(e, e') + F(e', b) - F(e, b)]_+, \tag{10}$$

where E_{tr} is a set of training events and $[x]_+ = \max(0, x)$. The

© 2020 Information Processing Society of Japan

difference between e and e', $\Delta(e, e')$, is 0 if e and e' are identical; otherwise, $\Delta(e, e') = \beta$. In our experiments, β was set to 0.3.

The pairwise ranking loss encourages a small (negative) value of $F(e', b) - F(e, b)$, especially when the difference between e and e' is large. In other words, when two events are dissimilar, $F(e, b)$ is expected to be much larger than $F(e', b)$.

When the dynamic search process is used in production, i.e., when no ground-truth relevance is known, the immediate reward of the query strategies are estimated using the relevant microblogs identified by the DNN-based model. We regard a microblog b as relevant to a given event e if the score $F(e, b)$ is sufficiently large among the scores for all events. An estimated set of relevant microblogs is formally described as

$$\hat{B}_t^+ = \{b \mid b \in B_t \wedge \mathrm{rank}(e, b) \leq r\}, \tag{11}$$

where $\mathrm{rank}(e, b)$ is the position of e when the events E are sorted by their scores F, and r is a hyper-parameter. The immediate reward is then computed by the same procedure as when the relevant microblogs are known *a priori*.

5. Experiments

Our experiments were intended to answer the following research questions: **RQ1**. Can our method effectively search for microblogs relevant to a given event? Is it also applicable to implicitly referred events? **RQ2**. Can our method outperform comparative methods, such as corpus-based and blog-to-event based methods? **RQ3**. Can our model identify relevant microblogs for a given event? Is it also applicable to events that are unseen in the training data?

5.1 Dataset Collection

For the microblog search experiments, we collected 28,406 tweets (microblogs) from Twitter, among which 14,203 tweets referred to a specific news (event) and the others were randomly selected. The collection procedure is detailed below:

(1) News (event) were collected from articles in CNN website [*1]). We collected news-like tweets posted by the news-feeder accounts ("@cnnbrk") in Twitter. We ensured that each news-like tweet m_e was unique by its tweet ID and linked to a news e in CNN website.

(2) Two human annotators checked the identity of collected news by their contents. As a result, there were total 108 unique news (no duplicate) and some of them were *consecutive* news, which is detailed below.

(3) We collected tweets that retweeted the news-like tweet m_e with a comment. These tweets were regarded as microblogs referring to news e.

We checked the identity of news by the following definitions. *News*: the circumstances about the subject of entities (e.g., people/object), for a specific time or place. *Consecutive news*: two or more news are considered as consecutive if their circumstances are different but used to describe the same subject. E.g., news 1 "Chris Brown and two other people arrested in Paris on allegations of aggravated rape and drug violations" and news 2 "Singer Chris Brown has been released from police custody and he won't face any charges at this time" were consecutive news as there are two different circumstances about the same subject ("Chris Brown").

Two annotators labeled the category of each news. Within each category, they then checked if each news is unique and if two or more news are consecutive news. **Table 2** shows the annotation results and "(4, 3, 2, 2)" in POLITICS means there were 4, 3, 2, and 2 news that were consecutive news, respectively.

Table 1 shows the statistics of the collected tweets. Implicit references are microblogs that refer to an event e without containing any terms in e, whereas explicit references contain terms that are also included in e. Clearly, our dataset contained more implicit references than explicit references. This trend supports our motivation of employing the DNN model for relevance estimation.

Table 1 Microblog statistics.

Data	#
Tweets	28,406
Tweets referring to events	14,203
Events (news)	108
Writers	24,107
Implicit references	9,137 (84.6 per event)
Explicit references	5,066 (46.91 per event)

Table 2 Event (news) annotation.

Category	Unique news: (total #:108)	Consecutive news
POLITICS	27	(4, 3, 2, 2)
CELEBRITY	26	(4, 3, 2, 2)
SOCIETY	18	
WORLD	10	(2)
CRIME	18	(6)
DISASTER	3	
TERROR	6	(3)

5.2 Microblogs, Events, and Search Engine

Microblogs are characterized by their textual content, the writer's profile [*2], a post location, and a post time. In our experiments, each microblog b was represented as a vector $\mathbf{b} = (\mathbf{b}_c, \mathbf{b}_u, \mathbf{b}_l, b_z)$. The content vector $\mathbf{b}_c$ was constructed as the weighted average of the word embeddings of the terms in the content

$$\mathbf{b}_c = \frac{1}{z} \sum_v \mathrm{TF\text{-}IDF}(v, b_c)\mathrm{WE}_c(v), \tag{12}$$

where v is each term of the content b_c, $\mathrm{TF\text{-}IDF}(v, b_c)$ indicates the weight of v in b_c, $\mathrm{WE}_c(v)$ indicates the vector of v in the word embeddings space for the contents, and $z = \sum_v \mathrm{TF\text{-}IDF}(v, b_c)$ is used for weighted average. The writer's profile vector $\mathbf{b}_u$ was constructed in the same way:

$$\mathbf{b}_u = \frac{1}{z} \sum_v \mathrm{TF\text{-}IDF}(v, b_u)\mathrm{WE}_u(v), \tag{13}$$

[*1] https://edition.cnn.com

[*2] Twitter allows each user to construct a profile describing their background, interests, and other distinguishing attributes.

© 2020 Information Processing Society of Japan

where v is each term of the writer's profile b_u and $\mathrm{WE}_u(v)$ indicates the vector of v in the word embeddings space for writer's profiles.

The location vector $\mathbf{b}_l$ was a two-dimensional vector indicating the latitude and longitude of the point from which the microblog was posted or the location of the writer's profile. Note that we preprocessed for microblogs that did not have posted locations in Appendix A.1. The post time b_z was simply represented by a scalar.

The vector of an event e was constructed in the same way above but used the word embeddings space WE_e for the event's contents, which is

$$\mathbf{e} = \frac{1}{z}\sum_{v} \text{TF-IDF}(v, e)\mathrm{WE}_e(v). \tag{14}$$

Note that we used different models for the word embeddings of microblogs and events, each of which was trained by a set of microblogs and a set of events, respectively, since the same term may have different senses in different media.

To simulate a search process, we built a simple search engine that returns the top-k microblogs according to the following matching score:

$$\sum_{i\in\{c,u,l,z\}} \alpha_i \mathrm{sim}_i(\mathbf{q}_i, \mathbf{b}_i). \tag{15}$$

k was set to 90 in the experiments. Note that we reference search engines for microblogs and use a complete form for query that comprises content terms, writer's profile terms, location, and time. The query vector $\mathbf{q} = (\mathbf{q}_c, \mathbf{q}_u, \mathbf{q}_l, q_z)$ was constructed in a similar way to the microblog vector:

$$\begin{aligned}\mathbf{q}_c &= \frac{1}{|V_c|}\sum_{v\in V_c} \mathrm{WE}_c(v),\\ \mathbf{q}_u &= \frac{1}{|V_u|}\sum_{v\in V_u} \mathrm{WE}_u(v),\end{aligned} \tag{16}$$

where v is either a query term for the contents or a query term for writer's profiles. The similarity function sim_i can be different. When $\mathbf{b}_i$ is either a content vector or a writer's profile vector, sim_i is the cosine of the two vectors. The similarity function computes the inverse of the Euclidean distance for location and time. The hyper-parameter α_i was tuned by the validation data.

5.3 Comparative Methods

We compared our method with the following methods from existing studies.

(1) **Corpus-based Search** (CW and CS) [8], [13], [35], [36], [37], [38], [39]: In this category of methods, the search queries are generated using the similarities and weights of the terms in event e, the search results B, and a corpus D. The first method (denoted by **CW**) uses the terms with the high frequency in the corpus and the latest search results to generate the next query $\mathbf{q}_t = (\mathbf{q}_{t,c}, \mathbf{q}_{t,u}, \mathbf{q}_{t,l}, q_{t,z})$:

$$\begin{aligned}\mathbf{q}_{t,c} &= \mathrm{w2v}\left(\arg\max_v \left(\lambda_B f_{B_{t-1}}(v) + \lambda_D f_D(v) + \lambda_n f_n(v)\right)\right),\\ \mathbf{q}_{t,u} &= \frac{1}{|B_{t-1}|}\sum_{b_i\in B_{t-1}} \mathbf{b}_{i,u},\\ \mathbf{q}_{t,l} &= \frac{1}{|B_{t-1}|}\sum_{b_i\in B_{t-1}} \mathbf{b}_{i,l},\\ q_{t,z} &= \frac{1}{|B_{t-1}|}\sum_{b_i\in B_{t-1}} b_{i,z},\end{aligned} \tag{17}$$

where $f_{B_{t-1}}(v)$, $f_D(v)$, and $f_n(v)$ are the frequency of word v in B_{t-1}, the frequency of v in D, and the inverse count of the queries that yielded search results including v, respectively. The function $\mathrm{w2v}(v)$ indicates the word embedding of v, and $\mathbf{q}_{t,u}$, $\mathbf{q}_{t,l}$, and $q_{t,z}$ are the means of the respective vectors in B_{t-1}.

The second method (denoted by **CS**) uses the same query vector as the CW, but a different $\mathbf{q}_{t,c}$. Intuitively, $\mathbf{q}_{t,c}$ is the vector representation of the most frequent term in the retrieved microblogs semantically similar to a given event e. In the CS, it is defined as

$$\begin{aligned}\mathbf{q}_{t,c} &= \mathrm{w2v}(\arg\max_v f_{B'_{t-1}}(v)),\\ B'_{t-1} &= \{b \mid b_i \in B_{t-1} \wedge \mathrm{sim}(\mathbf{e}, \mathbf{b}_{i,c}) \geq \theta_{\mathrm{CS}}\},\end{aligned} \tag{18}$$

where $\mathrm{sim}(\mathbf{e}, \mathbf{b}_{i,c})$, which should approximate the semantic similarity of e and b_i, is computed as the cosine of their vector representations trained by a corpus D. The hyper-parameter θ_{CS} is the threshold that determines whether a microblog is semantically similar to an event.

(2) **Blog-to-event-based Search** (B2E) [40], [41], [42]: This approach uses the same query vector as the CS, but a different $\mathbf{q}_{t,c}$. The main difference between CS and B2E is the similarity criterion that associates microblogs with a given event. As an event and its associated microblogs can use different vocabularies, this B2E approach first learns a mapping function from words of a microblog to words of an event in sequence, and computes the similarity between the mapped microblog and the event. Microblogs similar to an event e are defined as

$$B'_{t-1} = \{b \mid b_i \in B_{t-1} \wedge \mathrm{sim}(\mathbf{e}, \psi(b_{i,c})) \geq \theta_{\mathrm{B2E}}\}, \tag{19}$$

where ψ uses the sequence-to-sequence encoder-decoder model [43], [44], trained by a corpus D, which predicts the mapped words of a microblog and transforms the mapped words into a vector representation.

(3) **Stationary Strategy Search**: As a simplified version of our proposed method, this comparative method applies the patterns of query strategies to generate queries without considering the search state. There are the best sequence of query strategies (PS) and the best query strategy (SS).

(4) **Simplified Models for Ablation Study**: We also compared the performance of our proposed method with some simplification for an ablation study. In one setting of this experiment, RNN for the Q function was replaced by a method that considers only the current state and action (denoted by "No RNN"). The other setting is that we did not apply word embeddings for microblogs and events in DNN model that identifies relevant microblogs (denoted by "No WE").

5.4 Experimental Settings

This subsection explains the dataset setting, actions in the dynamic search process, training of the deep neural network models, and the baseline settings.

© 2020 Information Processing Society of Japan

5.4.1 Dataset Setting

Using the events and tweets shown in Table 1, the model is trained and evaluated as follows. We first split the set of events into E_{tr} and E_{test}, where $|E_{\text{tr}}| = 85$ and $|E_{\text{test}}| = 23$. To learn the model, we used the 80% of tweets referring to E_{tr} as the training data and the remaining 20% as the validation data. The number of tweets referring to E_{tr} was 12,547. The built model was evaluated for E_{test} by running the search processes on a search engine developed for the tweets referring to E_{test} (1,656 tweets) and the ones referring no events (14,203 tweets). Note that the events E_{test} was disjoint to the events E_{tr}, i.e., $E_{\text{test}} \cap E_{\text{tr}} = \emptyset$, and E_{test} is also called unseen events.

5.4.2 Query Strategies for the Dynamic Search Process

As mentioned, there were two types of query strategies for each feature of a microblog. Exploitative query strategies for a textual content and a writer's profile issue the term v that maximizes the following formula as a query at time $t + 1$:

$$\text{TF-IDF}(v, B_t)\text{TF-IDF}(v, B_t^+), \tag{20}$$

where the function TF-IDF(v, B_t) indicates the weight of v in B_t by the TF-IDF weighting schema. Recall that B_t is a set of microblogs retrieved at time t, and B_t^+ is a set of *relevant* microblogs retrieved at time t.

Exploitative query strategies for a post location and a post time issue the location or time of the microblog randomly sampled from the five microblogs closest to the mean post location or post time in search results B. By taking an example of the post time, the closeness of a microblog b_i is computed by:

$$|b_{i,z} - \bar{b}_z|, \tag{21}$$

where $\bar{b}_z$ is the mean post time in search results B_t. The closeness in terms of the post location is defined in a similar way with the Euclidean distance.

In contrast to the exploitative query strategies that encourage retrieval of microblogs similar to the latest search results, exploratory query strategies seek microblogs that are dissimilar to the latest search results. Exploratory query strategies for a content and a writer's profile issues the term v that maximizes the following formula as a query at time $t + 1$:

$$\text{TF-IDF}(v, B_t)^{-1}\text{TF-IDF}(v, B_t^+). \tag{22}$$

Only the difference from the exploitative query strategies is the inverse weight by TF-IDF(v, B_t). This is expected to encourage less frequent but salient terms to be issued as a query.

Exploratory query strategies for a post location and a post time issue the location or time of a microblog that is randomly sampled from the five microblogs farthest from the mean post location or post time in the search results B.

5.4.3 Tuning of the Dynamic Search Process

Table 3 summarizes our hyper-parameter settings in the dynamic search process. Below, we explain how we decided these parameters.

The training of the dynamic search process model was carried out with stochastic gradient descent with a learning rate 0.01. **Table 4** shows the effects of varying the length of search states for the Q function (τ), and the LSTM dimension (h), in the training of the models. The recall was maximized at $\tau = 6$ and $h = 50$.

Table 3 Hyper-parameter setting of our models.

		value
MDP	ϵ	0.15
	γ	[0.80, 0.99]
	Experience replay size	50
RNN	Sequence length (τ)	6
	LSTM dimension (h)	50
	# of trainable parameters of $\mathbf{W}$	12,608
DNN	event dimension (event's content)	216
	microblog's content dimension	216
	writer's profile dimension	216
	microblog dimension	435
	# of trainable parameters of $\mathbf{W}_e$ and $\boldsymbol{\delta}_e$	$216 \cdot 216 + 216$
	# of trainable parameters of $\mathbf{W}_b$ and $\boldsymbol{\delta}_b$	$216 \cdot 435 + 216$

Table 4 Tuning of the RNN model, showing the recall scores for different sequence lengths and LSTM dimension h.

RNN sequence length (τ)	LSTM's h h=**50**	h=30	h=60	h=100
3	0.672	0.644	0.659	0.616
4	0.729	0.712	0.724	0.708
5	0.729	0.717	0.721	0.661
6	**0.781**	0.762	0.772	0.708
7	0.776	0.751	0.767	0.707
8	0.755	0.745	0.747	0.719
9	0.714	0.697	0.709	0.667
10	0.599	0.575	0.584	0.538

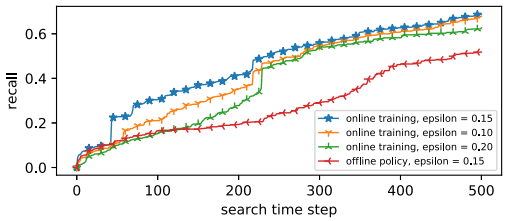


Fig. 3 Effect of online training for the dynamic search process (red plot is the result of no online training).

Compared with the static offline policy, our online optimized policy guided the search direction after acquiring some search experiences within 100 search steps (**Fig. 3**). Additionally, the search process might slightly favor random exploration of query strategies (i.e., $\epsilon = 0.15$). During each one-time training of the RNN model, we follow the early stopping of training as shown in the tuning of ϕ below.

We evaluated the dimension of the nonlinear transformation ϕ that was used for identifying relevant microblogs. We used stochastic gradient descent with a learning rate 0.01 of a decay rate 0.0001, and the pairwise ranking loss of Eq. (10). **Figure 4** showed the training loss and accuracy over epochs. It can be seen that the dimension 216 got better results among the other cases. We also decided to use the model parameters at epoch 150, as the top-5 accuracy 0.498 of the validation data did not improved with more epochs and the training loss was apparently low.

5.4.4 Baseline Methods

Using the training data, we obtained the frequency of terms, the word2vec model, and the blog-to-event sequence-to-sequence model. Furthermore, we ran search processes on the *validation* data to tune the hyper parameters including λ_B, λ_D, and λ_n of

© 2020 Information Processing Society of Japan

Eq. (17), θ_{CS} of Eq. (18), and θ_{B2E} of Eq. (19).

5.5 Experimental Results

We report and discuss experimental results of the microblog search and relevance estimation, and answer research questions from RQ1 to RQ3.

5.5.1 Performance of Microblog Search

Overall Performance

Table 5 shows the overall performances of our proposed method and comparative methods.

Our method (Ours) outperformed the baseline methods including the corpus-based search methods (CS and CW) and the blog-to-event-based search method (B2E). The recall was at least 1.67 times higher in our method than in the baseline methods. We can also observe that our method achieved the best recall in terms of searching for microblogs with implicit references. In our dynamic search process, applying word embeddings for relevance estimation was more effective than the one without using word embeddings (Ours (No WE)). In addition, applying dynamic query strategies (Ours) was more effective than the one applying stationary query strategies (Ours (PS)). Meanwhile, the corpus-based similarities (CS) and the blog-to-event mapping (B2E) were more effective for explicit references than using the important contents of high frequency (CW).

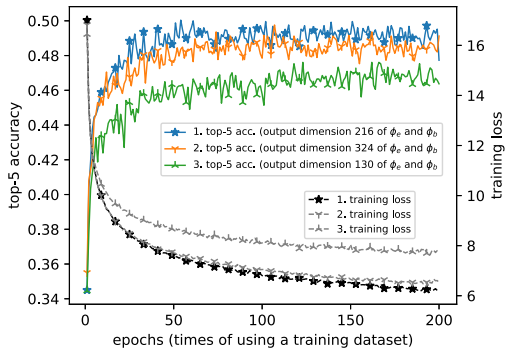


Fig. 4 Tuning of the DNN model.

Table 5 Search performance of each method.

Method	Recall	Implicit	Explicit
CS	0.413	0.238	0.175
B2E	0.389	0.192	0.197
CW	0.346	0.213	0.133
Ours	**0.688**	**0.361**	0.327
Ours (No RNN)	0.656	0.337	0.318
Ours (No WE)	0.492	0.254	0.238
Ours (PS)	0.576	0.336	0.240
Ours (SS)	0.499	0.283	0.216
Ours (Random)	0.322	0.177	0.145

These results are summarized as follows: (1) Our method effectively finds both implicit and explicit references to a given event. (2) The recall was improved not only by our dynamic query strategies but also by identifying the relevant microblogs. (3) The corpus-based similarities of word2vec and the blog-to-event mapping effectively retrieve certain types of explicit references, but are less effective for implicit references than our proposed method. (4) An appropriate mapping between the microblogs and the referred events is probably vital when searching for implicit references.

The research questions RQ1 and RQ2 are answered as follows: **RQ1**. Our method effectively searched for microblogs relevant to a given event, and also retrieved microblogs with implicit references to the event. **RQ2**. Our method outperformed the comparative methods (corpus-based and blog-to-event based methods).

Performance with Less Relevant Microblogs

As our dataset contained a fair amount of relevant microblogs, we also evaluated the performances with the dataset including fewer relevant microblogs, which might better represent the distribution of real-world microblog data. **Table 6** showed the recall of each method when a certain fraction of relevant microblogs were removed. Our method maintained its higher performance than the other methods, even when the relevant microblog ratio was reduced to 20%. However, the performance of implicit references decreased significantly. The reason might be that finding implicit references required more training data for learning the transformation used in the relevance estimation.

5.5.2 Performance of Relevance Estimation

Table 7 showed the performance of relevance estimation of microblogs. We ranked microblogs on a set of arbitrarily retrieved microblogs for a given event that was not used in the training. The ranking metrics is normalized discounted cumulative gain

Table 7 Performance of ranking microblogs for events.

Method	NDCG@5	NDCG@10	NDCG@60
CS	0.116	0.130	0.085
B2E	0.090	0.100	0.066
Ours	0.177	0.178	**0.236**
Ours (No loc./time)	0.168	0.167	0.221
Ours (No WE)	0.140	0.137	0.185

Table 6 Search performance of each method (evaluated on varying fractions of ground-truth relevant microblogs).

	80% of rel. microblogs			60% of rel. microblogs			40% of rel. microblogs			20% of rel. microblogs		
Method	Recall	Imp.	Exp.	Recall	Imp.	Exp.	Recall	Imp.	Exp.	Recall	Imp.	Exp.
CS	0.406	0.234	0.172	0.389	0.213	0.175	0.382	0.210	0.172	0.379	0.203	0.177
B2E	0.384	0.187	0.197	0.381	0.190	0.191	0.381	0.182	0.198	0.371	0.176	0.194
CW	0.343	0.209	0.133	0.333	0.198	0.134	0.324	0.190	0.133	0.311	0.180	0.131
Ours	0.643	0.312	0.331	0.624	0.296	0.328	0.615	0.292	0.323	**0.605**	0.277	0.328
Ours (No RNN)	0.606	0.292	0.314	0.602	0.284	0.318	0.587	0.270	0.317	0.570	0.271	0.300
Ours (No WE)	0.466	0.221	0.246	0.449	0.219	0.230	0.437	0.197	0.240	0.426	0.203	0.223
Ours (PS)	0.542	0.303	0.239	0.506	0.264	0.242	0.517	0.277	0.239	0.500	0.263	0.237
Ours (SS)	0.488	0.269	0.218	0.455	0.239	0.216	0.448	0.241	0.207	0.427	0.176	0.252
Ours (Random)	0.306	0.162	0.144	0.280	0.133	0.148	0.286	0.127	0.159	0.261	0.114	0.146

© 2020 Information Processing Society of Japan

Table 8 Case study of searching for microblogs referring to the event, "Suspect confesses to kidnapping Jayme Closs and killing her parents, and says he decided to target her after seeing ...". Note that we list queries used by a method that retrieved microblogs and "-" means no such queries. *Exploitative* query strategies for content, writer profile, post location, and post time are denoted by $a_{1,c}$, $a_{1,u}$, $a_{1,l}$, and $a_{1,z}$. Similarly, *Exploratory* query strategies for those items are denoted by $a_{2,c}$, $a_{2,u}$, $a_{2,l}$, and $a_{2,z}$.

	Microblog	Ref. type	CS	Ours (PS)	**Ours**
1	c:"Praying for this brave young woman." u:"Mid-century woman with decades of music in my head ..." l:"Florida" z:"Jan 14 20:53:24 2019"	Imp.	-	-	strategy $a_{2,u}$ q{c:"pray" "brave", u:"woman"}
2	c:"Turn this sicko into a twice baked potato. Poor girl." l:"Pittsburgh, PA" z:"Jan 14 20:50:33 2019"	Imp.	-	-	strategy $a_{2,l}$ q{c:"sicko", l:(close to Pittsburgh)}
3	c:"Sad and uncivilised moved by one from the civilised domains" l:"Gombe, Nigeria" z:"Jan 15 11:05:09 2019"	Imp.	-	strategy $a_{2,c}$ q{c:"sad" "uncivilised"}	strategy $a_{2,c}$ q{c:"sad" "civilised"}
4	c:"I know they always pointe out crimes committed by non-white and then tell us that they are not racist ..." u:"General News Politics History" l:"Bellingham, WA" z:"Jan 15 15:02:14 2019"	Imp.	q{c:"crime" "racist"}	strategy $a_{2,c}$ q{c:"crime" "racist"}	strategy $a_{1,u}$ q{c:"crime", u:"politics"}
5	c:"I wish this perv got killed in a shootout with cops! Poor little Jayme will now need to endure narrating ..." u:"UnCommented RT is an endorsement" z:"Jan 15 04:18:01 2019"	Exp.	q{c:"jayme" "kill"}	strategy $a_{1,c}$ q{c:"jayme"}	strategy $a_{2,c}$ q{c:"jayme" "poor"}
6	c:"Jayme you are a brave girl. He shall go to hell." u:"daughter. sister. mother. lover. friend. scrapbook event organizer. loves to laugh. loves to live." z:"Jan 14 22:31:42 2019"	Exp.	q{c:"jayme"}	strategy $a_{1,c}$ q{c:"jayme"}	strategy $a_{1,c}$ q{c:"jayme"}

(NDCG) with the ranking position of 5, 10, and 60.

We compared our proposed relevance estimation method (Ours) with simplified variants of our method and the baseline methods. The former includes the one without using the post location and time (Ours (No loc. and time)), and the one without using the representations of word embeddings (Ours (No WE)). The latter includes the corpus-based (CS) and the blog-to-event-based (B2E) methods.

The results suggested that: (1) The overall performance was low and this showed the difficultly of ranking on a set of randomly retrieved microblogs. (2) Our ranking was effective compared to other methods, especially if we were allowed to consider more ranking results, e.g., NDCG@60 = 0.236. (3) The mapping between representations of microblogs and events was effective than the mapping between word-to-word level of them. This implied that the referred events from microblogs was hard to model by considering only textual contents. (4) Both the representations of word embeddings and the transformation mapping was effective for identification of relevant microblogs. For high performance, the representations of microblogs should consider multiple attributes (e.g., textual content, writer's profile, location, time).

The research question RQ3 is answered as follows: **RQ3**. Our model could identify relevant microblogs for a given event more effectively than the baseline methods. It was also effective to events that are unseen in the training data.

5.6 Case Study

We conducted case study as shown in **Table 8**. Our methods made the effective query composed of text of non-keyword, such as sentiment words, cf. microblogs 1, 3, and 5. Our dynamic query strategy also constructed effective queries, for implicit references, such as those composed of related writer profiles or close locations, cf. microblogs 1, 2, and 4. In other words, stationary query strategies do not guarantee effective queries at the correct moment. Meanwhile, corpus-based search does not guarantee effective non-keyword queries. All of the tested methods constructed effective keyword queries, for explicit references, cf. microblogs 5, and 6.

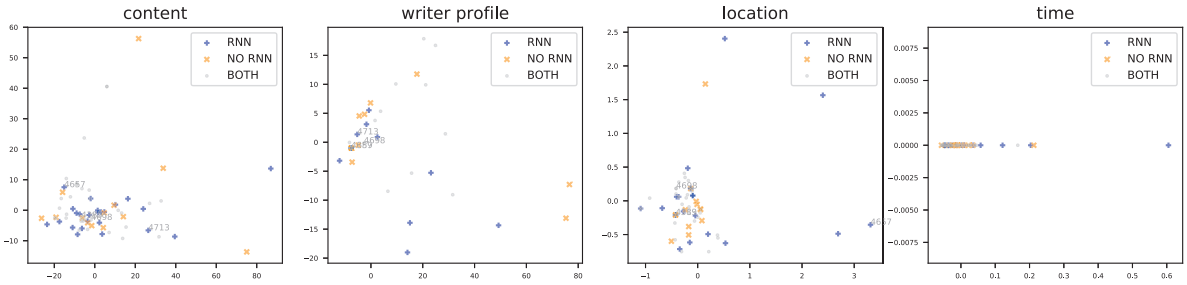
5.6.1 Effect of RNN Model

To investigate the effect of applying the RNN model in the dynamic search process, we compared the relevant microblogs found by the search processes that applied with the RNN model and without the RNN model (No RNN). We projected their results into two-dimensional spaces of textual contents, writer's profiles, locations, and times, respectively, by principal component analysis (PCA).

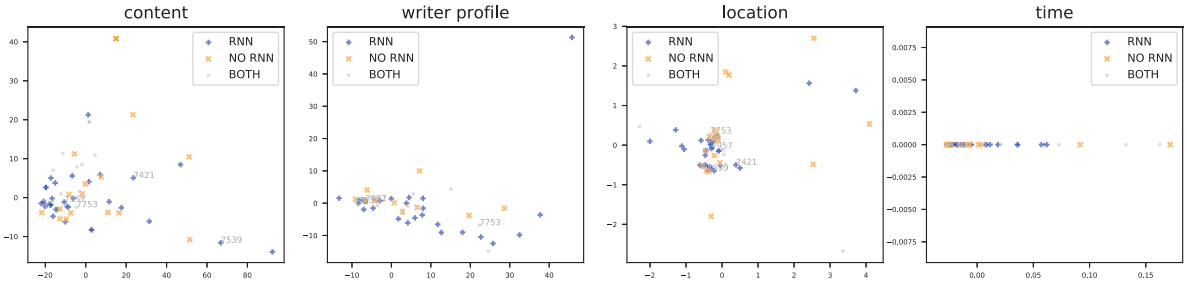
In **Fig. 5** (d), for the content space of event 209, most microblogs found by RNN or No RNN were overlapped as shown by the gray points. Meanwhile, these microblogs were in centers of microblogs and we could also find similar results on other events. This implied that either RNN or No RNN could find *representative* microblogs for events.

Furthermore, (1) RNN might find more *diverse* results, such as those of blue-cross points. In Figure (a), for event 68: "President Trump put the onus of an 18-day partial government shutdown on Democrats ...", RNN found a microblog that replied implicitly with negative emotion: "i can see the *pain* ...", compared to a

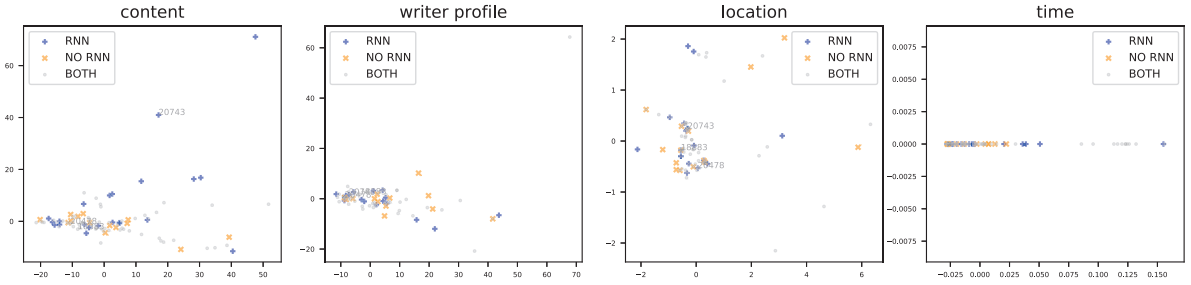
© 2020 Information Processing Society of Japan



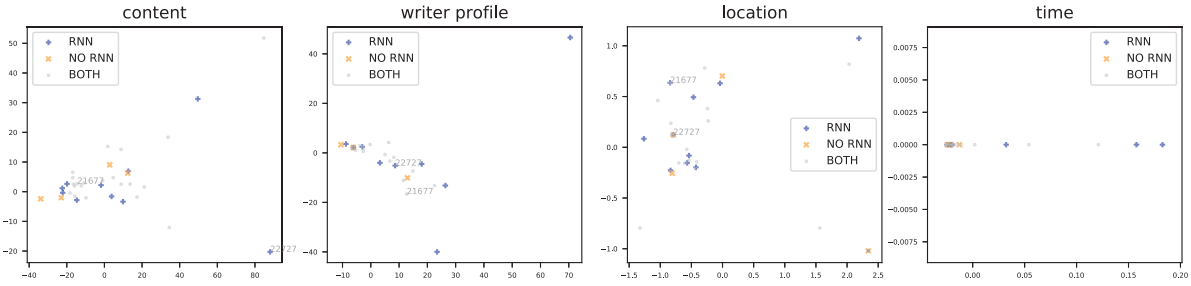
(a) Relevant tweets (microblogs) found for event 68: “President Trump put the onus of an 18-day partial government shutdown on Democrats, saying the matter ...”. A *center* tweet 4,698 (which was from the center of tweets): “But he was proud to take the mantle of a shutdown. Said he wouldn’t blame the Democrats. Lie?”. A *non-center* tweet 4,713: “i can see the pain. Use lube next time.”



(b) Relevant tweets found for event 100: “A mass drug overdose at a home in Chico, California, has killed one person ...”. A center tweet 7,753: “Good ole Mexican drugs in sanctuary Pelosi state of California”. A non-center tweet 7,421: “Love the people on here thinking a wall can stop America’s opioid addiction. Thats like hoping a rake can stop terrorism.”



(c) Relevant tweets found for event 180: “Multiple people have been shot at a SunTrust Bank in Sebring, Florida, authorities say ...”. A center tweet 18,883: “Really very sad news ... Now end the shooting ...”. A non-center tweet 20,743: “Build that wall.”



(d) Relevant tweets found for event 209: “In the midst of a partial government shutdown stalemate over a border wall, the State Department cancels a conference ...”. A center tweet 21,677: “Wanna see the Democrats end the shutdown and start supporting the unpaid federal employees?”. A non-center tweet 22,727: “This needs to stop.”

Fig. 5 Visualization of relevant microblogs found by search processes applying with RNN model and without RNN model. We projected found microblogs into two-dimensional spaces of textual contents, writer’s profiles, locations, and times, respectively. Blue cross, orange x, and gray point are a microblog found by RNN, without RNN, and both of RNN and without RNN, respectively. In the content space of Figure (b), for event 100: “A mass drug overdose at a home in Chico, California, has killed one person ...”, both models could find *representative* microblogs, from the center of microblogs, such as tweet 7,957: “Don’t do drugs, simple!!”; while only RNN was able to find more *diverse* microblogs, which might be far from the center results, such as tweet 7,421: “Love the people on here thinking a wall can stop America’s opioid addiction. Thats like hoping a rake can stop terrorism.”

© 2020 Information Processing Society of Japan

commonly found microblog: “But he was proud to take the mantle of a shutdown. Said he wouldn’t blame the Democrats”, which mentioned the keywords “shutdown” and “Democrats”. RNN also found more results of diverse writer’s profiles and locations, such as those in events 68 and 100. (2) Lastly, RNN exploited a bit more results that were around the center ones, such as in the content space of event 68.

The above results (1) and (2) might demonstrate the importance of *timely* exploratory or exploitative search with effective query features. Meanwhile, our RNN was effective to conduct *timely* exploratory and exploitative search based on the required historical search experiences, which might be the reason for the effectiveness of the RNN model.

6. Conclusion

We considered the problem of searching for microblogs referring to a particular event, which are difficult to find when the event’s contents are not explicitly stated in the microblog. In such cases, the searcher might not construct a suitable query for the search engine. To overcome the difficulty, we proposed an MDP-based dynamic search process that takes query strategies optimized for the current search state. As key components of the dynamic search process, we proposed an RNN-based model for predicting long-term returns of a search process, and a DNN-based model that tries to match between the representations of microblogs and those of events for identifying relevant microblogs.

To evaluate our proposed method, we conducted simulation-based experiments using 28,406 tweets sampled from Twitter. The experimental results suggested that (1) Our method could effectively search for microblogs relevant to a given event, especially for the implicitly referred events. (2) Our method could outperform comparative methods including corpus-based and blog-to-event-based methods. (3) Our model could identify relevant microblogs for a given event, from a set of microblogs. (4) Our models were applicable to events that were unseen in the training data.

Acknowledgments This work was supported by JSPS KAKENHI Grant Numbers JP16H02906, JP18H03244, JP18H03494, and JST PRESTO Grant Number JPMJPR1853, Japan.

References

[1] van Hasselt, H., Guez, A. and Silver, D.: Deep Reinforcement Learning with Double Q-Learning, *AAAI*, pp.2094–2100 (2016).
[2] Weiss, K.R., Khoshgoftaar, T.M. and Wang, D.: A survey of transfer learning, *J. Big Data*, Vol.3, p.9 (online), DOI: 10.1186/s40537-016-0043-6 (2016).
[3] Xia, L., Xu, J., Lan, Y., Guo, J., Zeng, W. and Cheng, X.: Adapting Markov Decision Process for Search Result Diversification, *SIGIR*, pp.535–544 (2017).
[4] Farag, M.M.G., Lee, S. and Fox, E.A.: Focused crawler for events, *Int. J. Digital Libraries*, Vol.19, No.1, pp.3–19 (2018).
[5] Klein, M., Balakireva, L. and de Sompel, H.V.: Focused Crawl of Web Archives to Build Event Collections, *WebSci*, pp.333–342 (2018).
[6] Gottschalk, S. and Demidova, E.: EventKG: A Multilingual Event-Centric Temporal Knowledge Graph, *ESWC*, pp.272–287 (2018).
[7] Gossen, G., Demidova, E. and Risse, T.: Extracting Event-Centric Document Collections from Large-Scale Web Archives, *TPDL*, pp.116–127 (2017).
[8] Farag, M.M.G.: Intelligent Event Focused Crawling, PhD Thesis, Virginia Tech, Blacksburg, VA, USA (2016).
[9] Zhang, C., Lei, D., Yuan, Q., Zhuang, H., Kaplan, L.M., Wang, S. and Han, J.: GeoBurst+: Effective and Real-Time Local Event Detection in Geo-Tagged Tweet Streams, *ACM TIST*, Vol.9, No.3, pp.34:1–34:24 (2018).
[10] Zhang, C., Liu, L., Lei, D., Yuan, Q., Zhuang, H., Hanratty, T. and Han, J.: TrioVecEvent: Embedding-Based Online Local Event Detection in Geo-Tagged Tweet Streams, *SIGKDD*, pp.595–604 (2017).
[11] François-Lavet, V., Henderson, P., Islam, R., Bellemare, M.G. and Pineau, J.: An Introduction to Deep Reinforcement Learning, *Foundations and Trends in Machine Learning*, Vol.11, No.3-4, pp.219–354 (2018).
[12] Han, M., Wuillemin, P. and Senellart, P.: Focused Crawling Through Reinforcement Learning, *ICWE*, pp.261–278 (2018).
[13] Odijk, D., Meij, E., Sijaranamual, I. and de Rijke, M.: Dynamic Query Modeling for Related Content Finding, *SIGIR*, pp.33–42 (2015).
[14] Jiang, L., Wu, Z., Feng, Q., Liu, J. and Zheng, Q.: Efficient Deep Web Crawling Using Reinforcement Learning, *PAKDD*, pp.428–439 (2010).
[15] Brittain, M., Bertram, J.R., Yang, X. and Wei, P.: Prioritized Sequence Experience Replay, *CoRR*, Vol.abs/1905.12726 (2019).
[16] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M.A.: Playing Atari with Deep Reinforcement Learning, *CoRR*, Vol.abs/1312.5602 (2013).
[17] Xian, Y., Lampert, C.H., Schiele, B. and Akata, Z.: Zero-Shot Learning - A Comprehensive Evaluation of the Good, the Bad and the Ugly, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.41, No.9, pp.2251–2265 (2019).
[18] Goodfellow, I., Bengio, Y. and Courville, A.: *Deep Learning*, MIT Press.
[19] Li, X., Guo, Y. and Schuurmans, D.: Semi-Supervised Zero-Shot Classification with Label Representation Learning, *ICCV*, pp.4211–4219 (2015).
[20] Lampert, C.H., Nickisch, H. and Harmeling, S.: Attribute-Based Classification for Zero-Shot Visual Object Categorization, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.36, No.3, pp.453–465 (2014).
[21] Rohrbach, M., Stark, M. and Schiele, B.: Evaluating knowledge transfer and zero-shot learning in a large-scale setting, *CVPR*, pp.1641–1648 (2011).
[22] Rohrbach, M., Stark, M., Szarvas, G., Gurevych, I. and Schiele, B.: What helps where - and why? Semantic relatedness for knowledge transfer, *CVPR*, pp.910–917 (2010).
[23] Zhang, Z. and Saligrama, V.: Zero-Shot Learning via Joint Latent Similarity Embedding, *CVPR*, pp.6034–6042 (2016).
[24] Changpinyo, S., Chao, W., Gong, B. and Sha, F.: Synthesized Classifiers for Zero-Shot Learning, *CVPR*, pp.5327–5336 (2016).
[25] Akata, Z., Reed, S.E., Walter, D., Lee, H. and Schiele, B.: Evaluation of output embeddings for fine-grained image classification, *CVPR*, pp.2927–2936 (2015).
[26] Wang, X., Ye, Y. and Gupta, A.: Zero-Shot Recognition via Semantic Embeddings and Knowledge Graphs, *CVPR*, pp.6857–6866 (2018).
[27] Xian, Y., Akata, Z., Sharma, G., Nguyen, Q.N., Hein, M. and Schiele, B.: Latent Embeddings for Zero-Shot Classification, *CVPR*, pp.69–77 (2016).
[28] Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Ranzato, M. and Mikolov, T.: DeViSE: A Deep Visual-Semantic Embedding Model, *NIPS*, pp.2121–2129 (2013).
[29] Zacks, J.M. and Tversky, B.: Event structure in perception and conception, *Psychological Bulletin*, Vol.127, No.1, pp.3–21 (2001).
[30] Liu, B., Kato, M.P. and Tanaka, K.: Cognition-Aware Summarization of Photos Representing Events, *IEICE Trans. Information and Systems*, Vol.99-D, No.12, pp.3140–3153 (2016).
[31] Gal, Y. and Ghahramani, Z.: A Theoretically Grounded Application of Dropout in Recurrent Neural Networks, *NIPS*, pp.1019–1027 (2016).
[32] Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R. and Schmidhuber, J.: LSTM: A Search Space Odyssey, *IEEE Trans. Neural Netw. Learning Syst.*, Vol.28, No.10, pp.2222–2232 (2017).
[33] Olah, C.: Understanding LSTM Networks (2015).
[34] Agarap, A.F.: Deep Learning using Rectified Linear Units (ReLU), *CoRR*, Vol.abs/1803.08375 (2018) (online), available from ⟨http://arxiv.org/abs/1803.08375⟩.
[35] Zamani, H. and Croft, W.B.: Relevance-based Word Embedding, *SIGIR*, pp.505–514 (2017).
[36] Mitra, B. and Craswell, N.: Neural Text Embeddings for Information Retrieval, *WSDM*, pp.813–814 (2017).
[37] Roy, D., Paul, D., Mitra, M. and Garain, U.: Using Word Embeddings for Automatic Query Expansion, *CoRR*, Vol.abs/1606.07608 (2016).
[38] Diaz, F., Mitra, B. and Craswell, N.: Query Expansion with Locally-Trained Word Embeddings, *ACL* (2016).
[39] Bendersky, M., Pueyo, L.G., Harmsen, J.J., Josifovski, V. and Lepikhin, D.: Up next: Retrieval methods for large scale related video suggestion, *KDD*, pp.1769–1778 (2014).
[40] Chen, W., Cai, F., Chen, H. and de Rijke, M.: Attention-based Hierar-

© 2020 Information Processing Society of Japan

chical Neural Query Suggestion, *SIGIR*, pp.1093–1096 (2018).
[41] Song, J., Xiao, J., Wu, F., Wu, H., Zhang, T., Zhang, Z.M. and Zhu, W.: Hierarchical Contextual Attention Recurrent Neural Network for Map Query Suggestion, *IEEE Trans. Knowl. Data Eng.*, Vol.29, No.9, pp.1888–1901 (2017).
[42] Sordoni, A., Bengio, Y., Vahabi, H., Lioma, C., Simonsen, J.G. and Nie, J.: A Hierarchical Recurrent Encoder-Decoder for Generative Context-Aware Query Suggestion, *CIKM*, pp.553–562 (2015).
[43] Chiu, C., Sainath, T.N., Wu, Y., Prabhavalkar, R., Nguyen, P., Chen, Z., Kannan, A., Weiss, R.J., Rao, K., Gonina, E., Jaitly, N., Li, B., Chorowski, J. and Bacchiani, M.: State-of-the-Art Speech Recognition with Sequence-to-Sequence Models, *ICASSP*, pp.4774–4778 (2018).
[44] Nallapati, R., Zhou, B., dos Santos, C.N., Gülçehre, Ç. and Xiang, B.: Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond, *CoNLL*, pp.280–290 (2016).

Appendix

A.1 Preprocessing for Microblogs' Locations

In real microblog data, such tweets, it often finds that microblogs were posted without attached locations. We preprocessed for the microblogs that did not have posted locations as follows.

For a microblog without posted location, (1) we used the location of the writer's profile instead. (2) If (1) did not work, we used a random location instead, which a random one from all existing posted or writer's locations.

The consideration of (2) is that since location was used as one factor that measured the similarity between microblogs and events, we must assign one value for the location. Meanwhile, we assign a random value for location to reduce the effect for the model that computes the similarity. In other words, suppose we did not assign the random but one specific value for those microblogs without location, then those microblogs would be considered as the same for their location values, which was not true in real world.

Jun-Li Lu received his M.S. degree in Electrical Engineering from National Taiwan University, Taiwan, in 2010, and received his Ph.D. degree in Informatics from Kyoto University, Japan, in 2020. His research interests include Information Retrieval and Machine Learning.

Makoto P. Kato received his B.S., M.S., and Ph.D. degrees in Informatics from Kyoto University, Japan, in 2008, 2009, and 2012, respectively. Since 2019, he has been an associate professor of Faculty of Library, Information and Media Science, University of Tsukuba. His research interests include Information Retrieval, Web Mining, and Machine Learning.

Takehiro Yamamoto received his B.S., M.S., and Ph.D. degrees in Informatics from Kyoto University, Japan, in 2007, 2008 and 2011, respectively. Since 2019, he has been an associate professor of School of Social Information Science, University of Hyogo. His research interests include Information Retrieval, Human-Computer Interaction, and Data Mining.

Katsumi Tanaka received his B.S., M.S., and Ph.D. degrees in Information Science from Kyoto University, Japan, in 1974, 1976 and 1981, respectively. In 1986, he became an associate professor of Department of Instrumentation Engineering, Faculty of Engineering, Kobe University. In 1994, he became a professor of Faculty of Engineering, Kobe University. In 2001, he became a professor of Graduate School of Informatics, Kyoto University. Since 2017, he has been an emeritus professor of Graduate School of Informatics, Kyoto University. His research interests include Database Theory and Systems, Web Information Retrieval, and Multimedia Retrieval.

(Editor in Charge: *Yuki Arase*)

© 2020 Information Processing Society of Japan