Representation Learning for Users’ Web Browsing Sequences∗

Yukihiro TAGAMI†, Member, Hayato KOBAYASHI†, Shingo ONO†, and Akira TAJIMA†, Nonmembers

SUMMARY Modeling user activities on the Web is a key problem for various Web services, such as news article recommendation and ad click prediction. In our work-in-progress paper [1], we introduced an approach that summarizes each sequence of user Web page visits using Paragraph Vector [3], considering users and URLs as paragraphs and words, respectively. The learned user representations are used among the user-related prediction tasks in common. In this paper, on the basis of analysis of our Web page visit data, we propose Backward PV-DM, which is a modified version of Paragraph Vector. We show experimental results on two ad-related data sets based on logs from Web services of Yahoo! JAPAN. Our proposed method achieved better results than those of existing vector models.

key words: online advertising, Web browsing behavior, Paragraph Vector, representation learning

1. Introduction

Large-scale Web sites that provide various Web services and mobile apps deal with a lot of user-related prediction tasks, such as news article recommendation [4] and ad click prediction [5], [6]. Feature engineering of user representations is very important to achieve high prediction accuracy for the tasks, but it is labor-intensive and inefficient for small-scale tasks. On the other hand, logs of user activities on the whole Web site are sufficiently available, such as Web page visits and search queries. For such cases, informative user representations, obtained via leveraging the history of user activities, are useful as features for the prediction tasks. In addition, low-dimensional feature vectors are preferable to high-dimensional sparse vectors for the tasks that have few training data. Figure 1 shows an overview of this approach.

Recently, in the natural language processing (NLP) field, distributed representations of words in a vector space have received much attention [7]. The studies that use this approach represent words as fixed length dense vectors, whereas the conventional approach treats individual words as unique symbols. These vector representations, which are learned by using various training methods, capture syntactic and semantic word relationships. In addition, some researchers have proposed models to learn vector representations for variable-length pieces of text such as sentences, paragraphs, and documents [3]. In a sentiment analysis task, this approach achieves better results than those of the conventional word n-gram model and simple averaging of word vectors.

Following these successful techniques, in our work-in-progress paper [1], we proposed an approach that summarizes each sequence of user Web page visits using Paragraph Vector [3], which is an unsupervised method that learns continuous distributed vector representations from pieces of text. In other words, we apply the vector model to sequences of user Web page visits, considering users and visits as paragraphs (or documents) and words, respectively. The learned low-dimensional vector representations of users are used among the user-related prediction tasks in common. For each prediction task, an individual classifier or regressor, such as logistic regression, is trained by using these common vectors as features and task-specific users’ properties or actions as targets. Note that we do not directly use Paragraph Vector to predict these targets, but utilize Paragraph Vector to create predictive fix-sized feature vectors from variable-length sequences.

However, do we simply treat the Web page visits data the same way as we treat natural language data? These two types of data are probably generated from different distributions. Therefore, in this paper, we first investigate the difference in the distribution between Yahoo! JAPAN access logs and English Wikipedia data. Then, on the basis of the difference, we propose Backward PV-DM, which is a mod-

Copyright © 2018 The Institute of Electronics, Information and Communication Engineers
ified version of Paragraph Vector. We report the extensive evaluations as well as the details of the improved methods.

Our main contributions are as follows.

- By comparing our Web page visits data with English Wikipedia data, we show the similarities and differences of frequency distributions between the two types of data. (Sect. 2.1)
- On the basis of the analysis of our Web page visits data, we propose Backward PV-DM. The difference between PV-DM and the proposed model is the context window. (Sect. 4)
- We evaluated our approach using two real-world data sets from an ad network and obtained better results than those of existing methods. (Sect. 5)

2. User Activities on the Web

We define \( A \) as a set of possible user activities that we consider. For an \( i \)-th user \( u_i \), the sequence of activities on the Web is also defined as \((a_{i,1}, a_{i,2}, \ldots, a_{i,T})\) where \( a_{i,t} \in A \) is the \( t \)-th activity of user \( u_i \), and \( T \) is the size of this sequence.

In this work, we focus on Web page visits and represent each visit \( a_{i,t} \) as a URL of the Web page. These URLs are just extracted from logs of Web services. Therefore, this method of representing the data is easy to use and scalable. Another option is to obtain hashed URLs that users have visited in the past via data partners in a similar way to the earlier studies \([8],[9]\) for targeting tasks in display advertising. Thus, our approach is simple and widely applicable. Since we represent each Web page visit as a URL, we use “Web page visit” and “URL” interchangeably. Table 1 shows examples of sequences.

Our approach can be easily extended to other types of events such as search queries and ad clicks. Therefore, we describe our approach using the generic activities \( a_{i,t} \) in Sects. 3 and 4.

2.1 Data Analysis on Web Page Visits

In this section, we reveal the difference between our Web page visit data and English Wikipedia data, since we apply an NLP-based approach to our data.

We collected part of the Yahoo! JAPAN access logs of July 22, 2014 and extracted the URLs of the Web pages that each user visited. These access logs included one of the mobile apps for smartphones and tablet computers as well as ordinary Web services. The data of users whose numbers of Web page visits were between 10 and 1000 were sampled. We discarded URLs that occurred fewer than five times in the extracted data. If the interval of time between two consecutive page visits exceeded 30 minutes, we considered that it was the start of a new session. A session in a sequence of Web page visits corresponds to a sentence in a paragraph or document. Consequently, there were about 3.87 million unique URLs and one billion page visits in the data.

For English Wikipedia data, we preprocessed the latest Wikipedia dump using Matt Mahoney’s script\(^1\) and the sentence segmenter in NLTK\(^1\).

In summary, we obtained two kind of observations by comparing the data.

- The frequencies of URLs in our Web page visit data follow a power-law distribution. The frequencies of words in English Wikipedia data have the same property, as is widely known\([11]\).
- By focusing on the relative position in a session or sentence, on the other hand, the two distributions of frequencies are significantly different.

The following part describes these two observations in detail.

First, the frequencies of URLs and words in the data are shown in Fig. 2. It is widely known that the frequencies of words in most languages follow a power-law distribution\([11]\). A power-law distribution looks like a roughly straight line in a log-log plot. Clearly, the plot of Web page visits shows as an approximately straight line\(^\dagger\). The exponent of the regression lines with power-law distribution are about \(-1.0\). The plot of the Wikipedia data seems to be a piecewise linear function. The exponents of the regression lines are \(-1.1\) for the early part of the data and \(-1.5\) for all of the data. Therefore, the frequencies in both data approximately follow a power-law distribution. However, the tail part of Web page visit data is “fatter” than that of English Wikipedia.

Next, the average of log frequency ratio for relative positions is shown in Fig. 3. The log frequency ratio for relative position \( k \), that is \( a_{i,t} \) and \( a_{i,t+k} \), is defined as follows:

\[
\log \left( \frac{\text{freq}(a_{i,t+k})}{\text{freq}(a_{i,t})} \right),
\]

where \( \text{freq}(a_{i,t}) \) represents the frequency of the Web page

\(^1\)http://mattmahoney.net/dc/textdata.html
\(^\dagger\)A straight line in a log-log plot is a necessary, but not sufficient, condition for the data following a power-law distribution\([11]\). Data generated by a log-normal distribution also looks roughly straight on the log-log plot.
Fig. 2 Log-log plot for Web page visit data (left) and English Wikipedia data (right). X-axis represents rank of activity or words in frequency table, and y-axis is number of occurrences. Solid and dashed lines represent regression lines for all data and early part of the data (rank less than $10^4$), respectively.

Fig. 3 Average of log frequency ratio for relative positions. Because of symmetric property of log ratio (y-axis) and relative position (x-axis), these plots are symmetric with respect to origin.

visit $a_{i,t}$ (or a word) in the data. We average the log frequency ratio of $t$ and $t+k$ in a session or sentence. The average values of English Wikipedia are around zero, which indicates that word frequencies do not change depending on the position in a sentence. By contrast, the average log frequency ratio of Web page visit data decreases as the relative position $k$ becomes larger. This suggests that URLs that appear in the latter part of a session are the “tail” URLs whereas the URLs that exist in the former part are the “head” URLs. This is caused by a trend of users’ Web browsing behavior. As shown in the examples in Table 1, most users of Yahoo! JAPAN visit the front page† at the beginning of the session and then follow the hyperlinks in the Web pages to move to different sites, such as news, sports, finance, and shopping. Similarly, on each site, users visit the Web pages that they are interested in by following the hyperlinks or using the search engine.

According to the above analysis, to capture the users’ interests or preferences suitably, the Web page visits of the “tail” URLs that appear in the latter part of the session are more important.

The above analysis is based on Yahoo! JAPAN access logs. However, we believe that there are access logs of other Web sites that have similar properties since most Web sites have a hierarchical structure similar to Yahoo! JAPAN. Our proposed approach in this study may work well on these sites. On the other hand, recently developed Web sites, such as social networking sites, show the personalized listing page that consists of other users’ posts and news articles recommended for each user. In this case, most users just move backward and forward between the listing page and posts. Thus, the access logs of these Web sites are regarded as having different properties, and other approaches may be suitable.

3. Existing Vector Models

In this section, we describe Paragraph Vector [3] and other vector models [7], [12] for our problem settings. We obtain the vector representations of users and URLs via learning these vector models with users’ URL sequences.

3.1 PV-DM

We first describe the PV-DM, Distributed Memory Model of Paragraph Vectors [3]. The objective of the vector model for an $i$-th user $u_i$’s sequence is to maximize the sum of log probabilities:

$$\sum_i \log p(a_{i,j} \mid a_{i,j-1}, \ldots, a_{i,j-s}, u_i),$$

where $s$ is the size of the context window. This means the conditional probability of the activity $a_{i,j}$ given preceding activities $a_{i,j-1}, \ldots , a_{i,j-s}$ and user $u_i$. The PV-DM defines the probability of this multi-class problem using the softmax function as follows:
where $w_{a_{i,t}}$ is the “output” vector corresponding to $a_{i,t}$, and $v_I$ is the “input” vector corresponding to the previous activities $a_{i,-1}, \ldots, a_{i,s-1}$ and user $u_i$. We also define the “input” activity vector corresponding to $a_{i,t}$ as $v_{u_i}$ and user “input” vector as $v_u$. Therefore, $v_I$ is represented as a concatenated vector:

$$v_I = [v_{u_i}^T, \ldots, v_{u_i}^T, v_{a_{i,t}}^T].$$

For the case of $j \leq 0$, an input activity vector $v_{a_{i,j}}$ is replaced with a special padding vector $v_{null}$. We define the size of input activity vector as $v_{a_{i,j}} \in \mathbb{R}^m$ and the size of input user vector as $v_{u_i} \in \mathbb{R}^n$, so both the size of input vector and output vector are represented as $v_I$, $w_{a_{i,t}} \in \mathbb{R}^{m+n}$. The overview of the model architecture is shown in Fig. 4 (left). The matrices $V_a$, $V_u$ and $W_a$ consist of the vectors $v_{a_{i,j}}$, $v_{u_i}$, and $w_{a_{i,t}}$, respectively. In other words, the vectors correspond to the row vectors in the matrices. We first randomly initialize the matrices, learn them, and then obtain the vectors by using a procedure described later in Sect. 4.2.

The user vector $v_{u_i}$ is used as a feature vector of various user-related prediction tasks, such as ad click prediction. We also use the “input” activity vectors $v_{a_{i,j}}$ as features and show the effectiveness in the experiment.

3.2 PV-DBoW

The PV-DBoW, Distributed Bag of Words version of Paragraph Vector, is another version of Paragraph Vector [3]. The objective of the PV-DBoW for an $i$-th user $u_i$’s sequence is to maximize the sum of log probabilities:

$$\sum_t \log p(a_{i,t} \mid u_i).$$

The probability of this multi-class problem is also defined using the softmax function as follows:

$$p(a_{i,t} \mid u_i) := \frac{\exp(w_{a_{i,t}}^T v_I)}{\sum_{a \in A} \exp(w_a^T v_I)}.$$  \hspace{1cm} (2)

For PV-DBoW, the input user vector $v_{u_i} \in \mathbb{R}^n$ and output word vector $w_{a_{i,t}} \in \mathbb{R}^m$ are the same size. The overview of the model architecture is also presented in Fig. 4 (right). PV-DBoW can be viewed as a simplified version of PV-DM where the size of the context window $s$ is zero.

3.3 CBoW and Skip-gram

For comparison with the above Paragraph Vectors, we also describe word vector models, CBoW and Skip-gram model[7].

Similar to Paragraph Vectors, the objective of CBoW and Skip-gram is also to maximize the sum of log probabilities, which is defined using the softmax function. However, these two vector models are proposed for obtaining word representation. Therefore, in our problem settings, these models just provide the representations for activities, not for users directly.

The objective function of the CBoW, Continuous Bag of Words model, is defined as follows:

$$\sum_t \log p(a_{i,t} \mid a_{i,-s}, \ldots, a_{i,-1}, a_{i+s+1}, \ldots, a_{i+s}).$$

$$p(a_{i,t} \mid a_{i,t-s}, \ldots, a_{i,-1}, a_{i,t+1}, \ldots, a_{i+t+s}) := \frac{\exp(w_{a_{i,t}}^T v_I)}{\sum_{a \in A} \exp(w_a^T v_I)},$$

where $v_I$ is the averaged vector of the context vectors:

$$v_I = \frac{1}{2s} \sum_{-s \leq k \leq 0} v_{a_{i,t+k}}.$$

On the other hand, the objective function of the Skip-gram model is as follow:

$$\sum_t \sum_{-s \leq k \leq 0} \log p(a_{i,t+k} \mid a_{i,t}).$$
conditional probability is defined as follows:

\[ p(a_{i+k} | a_{i}) := \frac{\exp(w_{a_i}^T v_{a_{i+k}})}{\sum_{a \in A} \exp(w_{a_i}^T v_a)} \] (3)

Figure 5 shows the overviews of these model architectures.

The Directed Skip-gram model proposed by Djuric et al. [12] is a modified model that considers the future activities given by the past activity:

\[ \sum_t \sum_{0 < k \leq 3} \log p(a_{i+k} | a_i). \]

### 4. Proposed Method

In this section, we propose Backward PV-DM. Then, we explain the learning method for these vector models.

#### 4.1 Backward PV-DM

On the basis of the analysis of our Web page visit data in Sect. 2.1, the Web page visits of “tail” URLs, which appear in the latter part of the session, are more important to capture the users’ interests or preferences suitably. Thus, we propose a modified model called Backward PV-DM. The difference between PV-DM and this model is the context window. The objective of the Backward PV-DM is to maximize the sum of log probabilities:

\[ \sum_t \log p(a_i | a_{i+1}, \ldots, a_{i+3}, u_i). \]

For predicting “output” activity \( a_i \), Backward PV-DM uses the following activities \( a_{i+1}, \ldots, a_{i+3} \) as “input” whereas PV-DM uses the previous activities \( a_{i-1}, \ldots, a_{i-3} \). The conditional probability is defined as follows:

\[ p(a_i | a_{i+1}, \ldots, a_{i+3}, u_i) := \frac{\exp(w_{a_i}^T v_f)}{\sum_{a \in A} \exp(w_{a_i}^T v_a)}, \] (4)

\[ v_f = [v_{a_{i+1}}^T, \ldots, v_{a_{i+3}}^T, v_u^T]. \]

The above modification of the context window encourages the storage of the information of URLs that appear in the latter part of a session in the user vector. Equation (4) is considered as a prediction for a URL of current time step \( a_i \) given user \( u_i \) and following URLs \( a_{i+1}, \ldots, a_{i+3} \). For the former part of a session, the prediction is relatively easier because future URLs are available. For example, in Table 1, a prediction of \( a_{2,2} \) given \( a_{2,3} \) is not difficult. Both URLs have the same domain (m. finance.yahoo.co.jp), and \( a_{2,3} \) is more specific than \( a_{2,2} \). Thus, user vector \( v_{u_i} \) does not need to memorize the information of \( a_{2,2} \). The user vector in the model acts as a memory that remembers what is missing from the current context. On the other hand, for the latter part of a session, the prediction is more difficult. For example, a prediction of \( a_{1,6} \) given \( a_{1,7} \) (Session end) has less information to predict \( a_{1,6} \). Therefore, user vector \( v_{u_i} \) needs to incorporate the information of \( a_{1,6} \) for the prediction. In this way, the modified objective function encourages the user vector to store the information of URLs that appear in the latter part of a session. On the other hand, plain PV-DM, which is based on Eq. (1), is likely to memorize the information of URLs in the former part of a session.

We also present Reverse PV-DM whose input sequences are just reversed, from future to past. Therefore, the conditional probability of Reverse PV-DM is the same as that of Backward PV-DM, but the sliding directions of the context window are different. The differences between PV-DM, Reverse PV-DM, and Backward PV-DM are summarized in Fig. 6.

The sliding direction of the context window does not change the whole objective to be maximized. However, this objective is not concave because of the bilinear form, and we search for a better local maximum of the objective using a stochastic gradient descent (SGD) as described in the following Sect. 4.2. In our implementation, the sliding direction is the same as the input order of the SGD procedure. Since the latter input is more memorable than the former input, the sliding direction and input order affect the quality of the user vector. In other words, the informative Web page visits that occur in the latter part of a session should be inputted last. The experimental results present the effect.

For comparison with Backward PV-DM, we use Back-
we use the “unigram” distribution $U(a)$ raised to the $3/4$th power as $p_n(a)$ in the same way as Mikolov et al. [7] did. We train the model using asynchronous SGD [13] with AdaGrad [14]. In the inference step for new users, the user vectors $v_u$ are learned while input and output activity vectors $v_t$ and $w_u$ are fixed.

4.2 Learning Vector Models

The computation of Eqs. (1)–(4) and their first derivative is impractical because the number of unique activities $|A|$ is typically large. Le and Mikolov [3] originally used hierarchical softmax with a Huffman binary tree based on word frequencies for fast training. Here, instead of hierarchical softmax, we use a negative sampling approach [7]. Hence an alternate objective to log $p(a_{i,t} \mid a_{i,t-1}, \ldots, a_{i,t-s}, u_i)$ with Eq. (2) is defined as:

$$\sum_{t} \sum_{t \leq j < 0} \log p(a_{i,t+j} \mid a_{i,j}) = \sum_{t=1}^{T_i} \sum_{0 \leq k < s} \log p(a_{i,t-k} \mid a_{i,t}).$$

The contextual ads displayed in AdClicker are determined by the Web page content as well as user information. Therefore, learning each Web page representation is also helpful for this task. On the other hand, SiteVisitor is the data set based on more active user interests.

The statistics for data sets are summarized in Table 2.

5.3 Proposed Methods and Baselines

We compared the methods using Paragraph Vector with some baselines. Bin and Freq are simple baselines that use raw URLs as features. Freq takes into account the frequencies of the user’s site visits, whereas Bin considers only whether a user visits the Web page or not (binary features).

Table 2: Statistics for two data sets. #Features is number of unique URLs that occurred more than or equal to five times in each data set.

| Data set    | #Train | #Validation | #Test  | #Features |
|-------------|--------|-------------|--------|-----------|
| AdClicker   | 51,576 | 10,000      | 10,000 | 66,957    |
| SiteVisitor | 1,862,693 | 20,000    | 20,000 | 1,219,850 |

5.1 Data Sets

We evaluated the proposed method using two supervised learning data sets: AdClicker and SiteVisitor. AdClicker consists of the users who clicked contextual ads that are included in the five selected ad campaigns. Similarly, SiteVisitor consists of the users who visited Web sites of the five selected advertisers.

In real application settings, sequences of user activities are typically summarized on a daily basis. Thus, we created these two data sets in view of predicting users’ particular actions on a day based on the history of Web pages visited the previous day. The training and validation sets were generated from logs of July 22 and 23, 2014. Web page visits on the former day are used as features, and the target activity in the latter day is treated as labels. Similarly, a test set was generated from logs of July 23 and 24, 2014, as features and labels, respectively. Since these features were extracted from Web service logs of Yahoo! JAPAN, they are only a small fraction of the entire user activities on the Web. These features do not include visits to advertisers’ sites, which are the labels of SiteVisitor, as well as ad clicks.

The contextual ads displayed in AdClicker are determined by the Web page content as well as user information. Therefore, learning each Web page representation is also helpful for this task. On the other hand, SiteVisitor is the data set based on more active user interests.

The statistics for data sets are summarized in Table 2.

The contextual ads displayed in AdClicker are determined by the Web page content as well as user information. Therefore, learning each Web page representation is also helpful for this task. On the other hand, SiteVisitor is the data set based on more active user interests.

The statistics for data sets are summarized in Table 2.

5.2 Evaluation Settings

AdClicker and SiteVisitor are multi-label data sets because a user can click more than one ad or visit various advertisers’ sites. In the experiment, we transformed the multi-label problem into a set of binary classification problems. We represent the binary classification tasks for AdClicker as Ac1 to Ac5 and for SiteVisitor as Sv1 to Sv5.

For each binary classification task, we trained logistic regression classifiers using features extracted by each method. The evaluation measure is Area Under ROC Curve (AUC).

5.3 Proposed Methods and Baselines

We compared the methods using Paragraph Vector with some baselines. Bin and Freq are simple baselines that use raw URLs as features. Freq takes into account the frequencies of the user’s site visits, whereas Bin considers only whether a user visits the Web page or not (binary features).
The feature vectors of these two methods are high dimensional sparse vectors.

We refer to CBoW, Skip-gram, Directed Skip-gram, and Backward Skip-gram as word vector models. We also refer to PV-DM, Reverse PV-DM, Backward PV-DM, and PV-DBoW as Paragraph Vectors. By using the word vector models, a user is represented as the simple averaging of input activity vectors $v_n$ in the sequence, which is similar to the approach of Djuric et al. [12]. We use the user vectors $v_n$ in Paragraph Vectors as user representations. These methods using the vector models are represented in italics. For example, the proposed method using the PV-DM model is represented as PV-DM.

For PV-DM and Backward PV-DM, we also use the averaging of input activity vectors $v_n$ in the same way as for the word vector models. We concatenated the user vectors and the averaged vector for the input of prediction tasks. These methods are called PV-DM(both) and Backward PV-DM(both). In addition, we evaluated a method that uses the concatenated vectors learned by the PV-DM and Skip-gram models. This method is called PV-DM + Skip-gram.

The settings of learning the vector models are as follows: the size of input vectors $v = v_n = 400$, the size of context window $s = 5$, the number of randomly sampled negative instances $k = 5$, and the number of epochs (full pass through the data) is five. For Paragraph Vectors, we create the user vectors $v_n$ via an inference step, considering all the users as new users. Because of the stochastic behavior of asynchronous SGD and random initialization, we report the mean value of five runnings for the methods using vector models.

The methods we described above are summarized in Table 3. Note that each vector model is learned by only using sequences of users’ activities extracted from logs of July 22, 2014. Therefore, any label information of the prediction tasks is not included in these models.

5.4 Results

The experimental results are summarized in Table 4. The bold elements indicate the best performance of the methods. The underlined scores are the best results of the word vector models and Paragraph Vectors.

The proposed Backward PV-DM achieved statistically significant improvements over PV-DM on almost all datasets. In addition, the results of Backward PV-DM are better than those of Reverse PV-DM. The difference between these models is just the sliding direction of the context window, in other words, the input order of the SGD procedure. Because of the stochastic behavior of asynchronous SGD and random initialization, we create the user vectors $v_n$ via an inference step, considering all the users as new users. Because of the stochastic behavior of asynchronous SGD and random initialization, we report the mean value of five runnings for the methods using vector models.

The methods we described above are summarized in Table 3. Note that each vector model is learned by only using sequences of users’ activities extracted from logs of July 22, 2014. Therefore, any label information of the prediction tasks is not included in these models.

### Table 3: Method summary

| Method              | Dimension | Density | #Models |
|---------------------|-----------|---------|---------|
| Bin                 | High      | Sparse  | –       |
| Freq                |           |         |         |
| CBoW                |           |         |         |
| Skip-gram           |           |         |         |
| Directed Skip-gram  |           |         |         |
| Backward Skip-gram  |           |         |         |
| PV-DM               |           |         |         |
| Reverse PV-DM       |           |         |         |
| Backward PV-DM      |           |         |         |
| BackwardPV-DM(both) |           |         |         |
| PV-DM(both)         |           |         |         |
| PV-DM + Skip-gram   |           |         |         |
| **PV-DM + Skip-gram** | 800       | Dense   | 2       |

### Table 4: Experimental results. Values are AUC. We report mean value of five runnings with different random initialization for methods using vector models (see Sect. 5.3 for more details).

|                  | AdClicker | SiteVisitor |
|------------------|-----------|-------------|
|                  | Ac1       | Ac2        | Ac3   | Ac4   | Ac5   | Sv1  | Sv2  | Sv3  | Sv4  | Sv5  |
| Bin              |           |            |       |       |       |       |       |       |       |       |
| Freq             | 0.9735    | 0.8063     | 0.6641| 0.7052| 0.7524| 0.7619| 0.8188| 0.7087| 0.7920| 0.7292|
| CBoW             | 0.9814    | 0.8184     | 0.6850| 0.6961| 0.7509| 0.7621| 0.8163| 0.7006| 0.7781| 0.7256|
| Skip-gram        | 0.9903‡   | 0.8323*    | 0.6531| 0.7154| 0.7700| 0.7999| 0.8277| 0.7067| 0.7849| 0.7339|
| Directed Skip-gram| 0.9906*   | 0.8354*    | 0.6562| 0.7163| 0.7725| 0.8017| 0.8328| 0.7135| 0.7931| 0.7417|
| Backward Skip-gram| 0.9904*   | 0.8374*    | 0.6533| 0.7159| 0.7706| 0.8019| 0.8308| 0.7120| 0.7914| 0.7394|
| PV-DM             | 0.9805‡   | 0.8328*    | 0.6525| 0.7138| 0.7712| 0.8018| 0.8307| 0.7125| 0.7909| 0.7388|
| Reverse PV-DM     | 0.9899    | 0.8151     | 0.6483| 0.7242| 0.7633| 0.8051‡| 0.8434‡| 0.7180‡| 0.7964‡| 0.7479‡|
| Backward PV-DM    | 0.9884‡   | 0.8263‡    | 0.6481| 0.7274‡| 0.7618| 0.8011‡| 0.8343‡| 0.7207‡| 0.7992‡| 0.7488‡|
| BackwardPV-DM(both)| 0.9902‡   | 0.8247‡    | 0.6537| 0.7345‡| 0.7661| 0.8092‡| 0.8366‡| 0.7222‡| 0.8028‡| 0.7491‡|
| PV-DM(both)       | 0.9894‡   | 0.8288‡    | 0.6507| 0.7290‡| 0.7581| 0.7965| 0.8294| 0.7198| 0.7945| 0.7489‡|
| PV-DM+Skip-gram   | 0.9910‡   | 0.8193     | 0.6351| 0.7379‡| 0.7704‡| 0.8134‡| 0.8373‡| 0.7220‡| 0.7998‡| 0.7506‡|
| BackwardPV-DM+Skip-gram| 0.9914‡   | 0.8231‡    | 0.6657‡| 0.7463‡| 0.7760‡| 0.8162‡| 0.8396‡| 0.7276‡| 0.8069‡| 0.7513‡|
| **PV-DM + Skip-gram**| 0.9912‡   | 0.8358*    | 0.6622*| 0.7391*| 0.7752*| 0.8128*| 0.8396*| 0.7252*| 0.8026*| 0.7531*|
Fig. 7 Experimental results when changing train data size. Horizontal axis (train data size) is logarithmic scale. Due to space limitations, we only show results of Ac2 (AdClicker), Sv2, and Sv5 (SiteVisitor).

Fig. 8 Experimental results when changing the number of Web site visits in learning the vector models. Horizontal axis (number of Web site visits) is logarithmic scale. Due to space limitations, we only show results of Ac2 (AdClicker), Sv2, and Sv5 (SiteVisitor).

ested in them. Thus, learning Web page representations \( v_{u_t} \) is also helpful for these tasks. On the other hand, Paragraph Vectors consistently demonstrated superior performance on the SiteVisitor dataset. This dataset is just based on users’ active interests since users visit the advertisers’ Web sites by their own will. Therefore, Paragraph Vectors achieved better results. Note that the proposed Backward PV-DM achieved statistically significant improvements over PV-DM on the AdClicker dataset as well. Thus, our proposed improvement for PV-DM, which was inspired by data analyses, worked effectively.

Backward PV-DM(both) achieved the best results in seven of ten tasks. Since this method utilizes both user and URL vectors obtained by Backward PV-DM training, it achieved statistically significant improvements over Skip-gram, even on three of five AdClicker tasks. As described above, Skip-gram achieved better results than plain Backward PV-DM for this dataset. Backward PV-DM(both) is based on the vectors obtained from Backward PV-DM whereas PV-DM + Skip-gram needs to train both PV-DM and Skip-gram models. Thus, Backward PV-DM(both) is easy to train.

To summarize the above results, Backward PV-DM showed superior performance to plain PV-DM, and Backward PV-DM(both) which utilizes both user and URL vectors achieved the best results in most tasks.

5.5 Effect of the Data Size

As described in Sect. 1, we focus on achieving better prediction accuracy for the small-scale tasks, leveraging the history of user activities. The experimental results when changing train data size of AdClicker and SiteVisitor are shown in Fig. 7. Compared with Bin and Freq, whose features are high-dimensional sparse vectors, the vector models achieve the better results when the size of training data is small. Since the vector models learn the low-dimensional dense vector, they have an advantage in this case.

We also evaluate whether the prediction accuracy is improved when the size of Web page visit data is bigger. Figure 8 shows the experimental results when changing the number of Web page visits in learning the vector models. The results of some tasks show the improvements. In other words, the prediction accuracies for user-related tasks can be improved by simply increasing the data size in learning the vector models. The degree of the improvements is relatively small. However, since the learned user representations are used among the user-related prediction tasks in common, it is not a small effect for all small-scale tasks.

6. Related Work

In the online advertising field, some previous works focused on finding the user segments that might be interested in given advertiser’s products, inferred from web-browsing behavior information. These approaches are known as behavioral targeting [15] or conversion optimization [9], [16]. Advertisers increase the effectiveness of advertising to deliver their ads to the audience found by the approaches.

Perlich et al. [9] presented a transfer learning approach...
for online display targeting. In the first stage of the approach, users are represented as a bag-of-words representation of the users browsing history, with each URL hashed into its own binary feature. The Bin method, which is compared with the proposed method in Sect. 5, is similar to this approach.

Djuric et al. [12] proposed an approach for improving estimation of ad click or conversion probability on the basis of a sequence of a user’s online actions modeled using the Hidden Conditional Random Fields (HCRF) model [17]. To address the sparsity issue at the input side of the HCRF model, the authors proposed a directed version of the Skip-gram model, which maximizes the log-probabilities of future activities given users’ preceding activities. Input “words” of the Directed Skip-gram model consist of entities found on a Web page visited by the user and tokens in search queries.

Okura et al. [4] tackled news article recommendation tasks. They generated user representations by using Recurrent Neural Networks (RNNs) with browsing histories as input sequences. Therefore, this approach is similar to ours. However, they learned task-specific RNNs by using users’ clicks as labels whereas we simply learned common vector models for various user-related prediction tasks in an unsupervised manner.

White et al. [18] focused on Web site recommendations and studied the effectiveness of various sources of contextual information for user interest modeling. Similar to ours, their method modeled user interests by using the URLs that the user visited. However, they represented each Web page as pre-defined Open Directory Project (ODP) categories by using trained classifiers, and aggregated these categories are regarded as user interest. On the other hand, we obtained both the URL and user vectors simultaneously by learning the vector models from sequences of user visits without supervision. In addition, they just aimed at recommending Web pages to each user whereas we intend to use learned low-dimensional vectors among the user-related prediction tasks in common.

Yan et al. [19] studied Web caching and prefetching. They improved these policies by mining frequent access patterns of Web documents and building association-based prediction models. Therefore, their objective is predicting users’ upcoming Web accesses. On the other hand, our study focuses on extracting common user representations from sequences of Web page visits via prediction.

For an English to French translation task, Sutskever et al. [20] reported that the reversed input of the words in the source sentence when using a Long Short-Term Memory (LSTM) model achieved the better result. This technique is related to our discussion of the difference between Reverse PV-DM and Backward PV-DM.

For obtaining continuous word representations, Bojanowski et al. [21] proposed an approach where each word is represented as a bag of character n-grams. A vector representation of each character n-gram is learned using a large corpus, and each word is represented as the sum of these vectors. The authors showed that their word vectors achieved state-of-the-art performance on several word similarity and analogy tasks. This idea may be applied to our approach by representing each URL as a bag of substrings, such as domain names.

7. Conclusion

In this paper, on the basis of the analysis of our Web page visit data, we proposed Backward PV-DM, which is a modified version of Paragraph Vector. We evaluated this approach on two ad-related data sets based on logs from Yahoo! JAPAN Web services. The experimental results demonstrated the effectiveness of our proposed method.

References

[1] Y. Tagami, H. Kobayashi, S. Ono, and A. Tajima, “Modeling user activities on the web using paragraph vector,” Proceedings of the 24th International Conference on World Wide Web, pp.125–126, 2015.
[2] Y. Tagami, H. Kobayashi, S. Ono, and A. Tajima, “Distributed representations of web browsing sequences for ad targeting,” The 2nd International Workshop on Ad Targeting at Scale, 2016.
[3] Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” Proceedings of the 31st International Conference on Machine Learning, pp.1188–1196, 2014.
[4] S. Okura, Y. Tagami, S. Ono, and A. Tajima, “Embedding-based news recommendation for millions of users,” Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.1933–1942, 2017.
[5] H. Cheng and E. Cantu-Paz, “Personalized click prediction in sponsored search,” Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp.351–360, 2010.
[6] H.B. McMahon, G. Holt, D. Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A.M. Hrafnikelsson, T. Boulos, and J. Kubica, “Ad click prediction: A view from the trenches,” Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.1222–1230, 2013.
[7] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” Advances in Neural Information Processing Systems 26, pp.3111–3119, 2013.
[8] B. Dalessandro, D. Chen, T. Raeder, C. Perlich, M. Han Williams, and F. Provost, “Scalable hands-free transfer learning for online advertising,” Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.1573–1582, 2014.
[9] C. Perlich, B. Dalessandro, T. Raeder, O. Stitelman, and F. Provost, “Machine learning for targeted display advertising: transfer learning in action,” Machine Learning, vol.95, no.1, pp.103–127, 2014.
[10] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python, O’Reilly Media, Inc., 2009.
[11] A. Clauset, C.R. Shalizi, and M.E.J. Newman, “Power-law distributions in empirical data,” SIAM review, vol.51, no.4, pp.661–703, 2009.
[12] N. Djuric, V. Radosavljevic, M. Grbovic, and N. Bhamidi, “Hidden conditional random fields with distributed user embeddings for ad targeting,” IEEE International Conference on Data Mining, pp.779–784, 2014.
[13] B. Recht, C. Re, S. Wright, and F. Niu, “Hogwild: A lock-free approach to parallelizing stochastic gradient descent,” Advances in Neural Information Processing Systems 24, pp.693–701, 2011.
[14] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods
for online learning and stochastic optimization,” Journal of Machine Learning Research, vol.12, no.Jul, pp.2121–2159, 2011.

[15] M. Aly, A. Hatch, V. Josifovski, and V.K. Narayanan, “Web-scale user modeling for targeting,” Proceedings of the 21st International Conference on World Wide Web, pp.3–12, 2012.

[16] Y. Liu, S. Pandey, D. Agarwal, and V. Josifovski, “Finding the right consumer: Optimizing for conversion in display advertising campaigns,” Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, pp.473–482, 2012.

[17] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell, “Hidden conditional random fields,” Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol.29, no.10, pp.1848–1852, 2007.

[18] R.W. White, P. Bailey, and L. Chen, “Predicting user interests from contextual information,” Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.363–370, 2009.

[19] Q. Yang, H.H. Zhang, and T. Li, “Mining web logs for prediction models in www caching and prefetching,” Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.473–478, 2001.

[20] I. Sutskever, O. Vinyals, and Q.V. Le, “Sequence to sequence learning with neural networks,” Advances in Neural Information Processing Systems 27, pp.3104–3112, 2014.

[21] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” arXiv preprint arXiv:1607.04606, 2016.