Abstract—Many prediction models have been proposed to improve the effectiveness of web prefetching for reducing the response time perceived by users when browsing the web. Most of these models are based on structure learning and are applied at the client side. Currently, considerable attention is being paid to proxy-based prefetching because it is more effective and accurate in predicting the correlated pages of many websites of similar interest for more homogeneous users. Compared with client-based prefetching, more complex prediction tasks must run in the proxy, which implies that a more powerful prediction model is required. Thus, based on the time-series characteristics of browsing records, we proposed the intentionality-related long short-term memory (Ir-LSTM) model, which combines both the Skip-Gram embedding method and the LSTM model while expanding the input features with user information. We also propose a novel dynamic allocation module for detecting real-time traffic bursts and correspondingly adjusting the correlation coefficient of the model’s output to achieve higher server-side resource utilization while fully maximizing hit ratio.

Index Terms—Web prefetching, Web prediction model, Intentionality-related long short-term memory (Ir-LSTM)

I. INTRODUCTION

Web prefetching can predict and prefetch relevant web objects that users may visit in the near future by considering their access patterns, thereby effectively reducing user-perceived latency. Different prediction models such as web-link analysis [1], [2], which refers to the structure of the web page and the frequency distribution of the user’s browsing records, have been proposed, but few studies have investigated time-series inputs based on users’ prior access frequency. Recent structure-learning-based methods consider the time-series historical records of users’ web visits, and methods based on a Markov chain model, hidden Markov model (HMM), and graph are widely used for time-series inputs [3]. However, the different types of users and webpages browsed complicate the prediction task on proxy servers.

This study develops an intentionality-related long short-term memory (Ir-LSTM) model as a prediction method. The model input integrates both webpages’ and users’ information (e.g., webpage title, IP address, and time). To achieve a higher resource utilization, the prediction model only outputs valuable links having large size or high latency. A dynamic allocation module including a real-time traffic burst detector and a dynamic allocation mechanism is developed to correspondingly adjust the output parameters, namely, the number of links in the output vector that include the prediction probability for each link and the skipping threshold. In this module, a bloom filter and count-min (CM) sketch methods are used for real-time detection and calculation of streaming data. By using this dynamic allocation module, the prediction model not only maximizes the resource utilization to achieve a higher hit ratio but also ensures that the server will not be overloaded.

II. IR-LSTM

Before training the model, all output labels are divided into two categories: prefetching and skipping. All links in the datasets are filtered by their size and latency. Links larger than 50 KB or with latency higher than 1.5 s are added to the prefetching list whereas others are labeled with “skip.”

A. Prediction Model

Fig. 1 shows the schematic of our Ir-LSTM module. The time-series inputs of accessed information are divided into three parts: URL, IP/timestamp, and webpage title. First, the URL is encoded via a Skip-Gram layer, which is one of the embedding structures in Word2Vec [4]. This embedding enables quantitative measurements of the relationship between links. After training the Skip-Gram neural networks, the current input URL is encoded into vector $V_i$ by mapping via its hidden layer matrix $W$. Then, $V_i$ is provided to the LSTM module, where a cell unit set is used to store the calculated value of the former input. The LSTM module has three types of gates, which are used to control the memory, forgetfulness, and output ratio of the current information. Finally, the current output $h_t$ is obtained.
The other two inputs are processed through different embedding layers. Preprocessing steps such as geolocating the IP address and mining and clustering (k-nearest neighbors) methods for topic classification of the title are also implemented. Then, the new embedded vectors are added to the output of the LSTM module $h_t$ to form the input vector $V ([h_t, V_{time}, V_{IP}, V_{type}])$ of a two-layer neural network model. The model finally outputs a probability distribution that each link will be prefetched and the skip probability, using a SoftMax function (see Fig. 1).

### B. Dynamic Allocation Module

In the dynamic allocation module, we use a Bloom filter to monitor the accessing load. The bloom filter returns whether each link is in the prefetching list or not (1 or 0, respectively). The load $C_t$ at time interval $t$ is calculated by (1), where $S_i$ is the size of link $i$, and $n_{l,t}$ is the number of links at time interval $t$. The allocation trigger is activated when the gradient vector $|\nabla C_t|$ is greater than the given burst threshold.

$$C_t = \sum_{i=1}^{n_{l,t}} S_i \ast \text{Bloom Filter}(\text{link}_i).$$  

Once activated, the repetitive rate is calculated based on the CM sketch algorithm, which can output the frequency of each link in the prefetching list or not. For all links in the list, the repetitive rate is $\eta = B/M$, where $M$ is the total frequency obtained from all results of the CM sketch and $B$ is the sum of the return values of the bloom filter. The trained Ir-LSTM model is used to search for the most appropriate variables [$n_{pf}$ (number of links to be prefetched) and $\sigma_s$ (skipping threshold)] in the testing set that can not only reach the maximum hit ratio but also satisfy (2).

$$n_{pf} \times \eta \times F_S (\sigma_s) \times C_t < R_{ub},$$  

where $F_S$ is a mapping function that returns the skipping ratio of the model and $R_{ub}$ is the resource upper bound.

### III. Preliminary Experiments

For the implementation, dropout and regularization methods are used in the training process to overcome the overfitting problem. The Adam optimizer is selected for the gradient descent process, and the gradient clipping method is used for the exploding gradient problem. The data used in this article is taken from HTTP requests from different users at Boston University (BU) in 1995. The link information of each article is taken from HTTP requests from different users at Boston University (BU) in 1995. The link information of each article is taken from HTTP requests from different users at Boston University (BU) in 1995. The link information of each article is taken from HTTP requests from different users at Boston University (BU) in 1995.

In future work, a gradient ascent method will be considered to find the best parameter set more quickly without traversing all results in the testing set. Further, the prediction module can be further improved by mixing both structure-learning and deep-learning methods.

### REFERENCES


