Segmenting Search Query Logs by Learning to Detect Search Task Boundaries

Luis Eduardo Lugo Martinez, Jose G. Moreno, Gilles Hubert

Segmenting Search Query Logs by Learning to Detect Search Task Boundaries

Luis Lugo
IRIT UMR 5505 CNRS, U. de Toulouse
Toulouse, France
luis.lugo@irit.fr

Jose G. Moreno
IRIT UMR 5505 CNRS, U. de Toulouse
Toulouse, France
jose.moreno@irit.fr

Gilles Hubert
IRIT UMR 5505 CNRS, U. de Toulouse
Toulouse, France
gilles.hubert@irit.fr

ABSTRACT

To fulfill their information needs, users submit sets of related queries to available search engines. Query logs record users' activities along with timestamps and additional search-related information. The analysis of those chronological query logs enables the modeling of search tasks from user interactions. Previous research works rely on clicked URLs and surrounding queries to determine if adjacent queries are part of the same search tasks to segment the query logs properly. However, waiting for clicked URLs or future adjacent queries could render the use of these methods unfeasible in user supporting applications that require model results on the fly. Therefore, we propose a model for sequential search log segmentation. The proposed model uses only query pairs and their time span, generating results suited for on the fly user supporting applications, with improved accuracy over existing search segmentation approaches. We also show the advantages of fine-tuning the proposed model for adjusting the architecture to a small annotated collection.

KEYWORDS

Information need; Search task segmentation; Recurrent neural network

ACM Reference Format:
Luis Lugo, Jose G. Moreno, and Gilles Hubert. 2020. Segmenting Search Query Logs by Learning to Detect Search Task Boundaries. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3397271.3401257

1 INTRODUCTION

Search engines are an essential component of the interactions of users with the World Wide Web. They are crucial to help users access the ever-increasing amount of information available. A wide range of needs and desires from users are converted to queries and submitted to available search engines [8]. Both automatic and manual analysis of search interactions from users enables the modeling of users' search patterns. Extracted patterns help to personalize search engine interactions. In turn, personalization allows the modification of search results depending on the user, the retrieval of advertisement according to user interests, the suggestion of queries related to user information needs, and other tasks designed to support the user while she performs her search tasks [3, 8].

Search engine logs register the queries users run in the search engines to complete their search tasks. Mining those logs allows the identification of search tasks. Search session segmentation is the first step in multiple methods for search task identification[8, 10, 12]. In session segmentation, a sequential log of search queries is partitioned into smaller sequences of queries. The boundaries for the query log partitions lie in pairs of adjacent queries. To determine if a query pair is a boundary, one may use time spans between the queries [7, 10, 12, 18]. If the time span is larger than a certain threshold, the query pair is considered a session boundary, which means that each query belongs to a different session. More sophisticated approaches use heuristics-based models [5] or neural networks [3] to determine the boundaries in the sequential query log. However, the use of clicked URLs [5] or adjacent queries [3] in the search log - in both backward and forward directions - represents a limitation in practical setups, especially in user supporting applications that require modeling on the fly. In such applications, waiting for the clicked URL or future queries to populate the model input might be unfeasible.

Hence, we propose a bidirectional recurrent neural network (RNN) architecture that segments pairs of adjacent queries based on semantic representations of queries and time spans between queries to provide temporal information. The segmentation model determines if adjacent pairs of queries represent a boundary between search tasks, without relying on clicked URLs, character-level representations, or surrounding queries for optimum performance. Furthermore, we test the segmentation architecture in a fine-tuning setup [1] to know how the model adapts to a small log dataset.

2 RELATED WORK

To support users during the information seeking process, it is crucial to segment query logs correctly according to their search tasks. Some methods based on time spans between queries have been proposed to extract task-based sessions from query logs [10]. Also, search log segmentation based on time is part of several methods [5, 7, 10, 15, 18]. However, when analyzing search logs, multiple information needs overlap because users tend to solve various search problems during their interactions with the search engine [7, 8, 10]. Because of this multitasking, a time-based session might contain more than one task session [10, 15]. Still, query timestamp information is essential. It provides a chronological structure to

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401257

the query logs. Bestlink SVM [18] leverages that chronological structure by establishing links only with backward queries.

Neural networks have also been applied to session segmentation in query logs. Context Attention based Long Short-Term Memory (CA-LSTM) [3] determines if two adjacent queries pertain to a session boundary or not. Three sequences encode the data, in contrast with other models [9], where a single input encodes data, and a bidirectional RNN processes the single input to produce the model results. Moreover, exploiting the context from query logs in CA-LSTM is necessary to generate an improvement over the existing methods and surpass its baseline approach [12], which is intended to extract the hierarchical structure of tasks and subtasks embedded in search logs. A heuristics-based session segmentation method [7] also models search logs as hierarchies of search tasks. The heuristics-based session segmentation method (HBSSM) [5], an improved heuristics-based method, leverages pre-trained word embeddings to provide semantic similarity measures to segment the query logs, complementing semantic relatedness with temporal, lexical, and clicked URLs heuristics. Thus, a cascade of heuristics is applied to each query pair, and manually set parameters provide thresholds in each heuristic to generate the output of the model.

However, there are some limitations in existing approaches. Graph-based clustering methods [10, 18] do not scale well, making the processing of search logs computationally expensive as the size of the query log grows. Also, heuristics-based methods [5, 7] have several manually set thresholds in their rules, making it challenging to adapt automatically to other labeled datasets if needed. Likewise, the need for clicked URLs [5] or adjacent queries to properly segment search logs [3] could be problematic. Some user supporting applications need on the fly results; thus, they can not wait for clicked URLs or future user queries to provide forward context. Also, when dealing with simple search tasks like fact-finding, typically, a single query could solve the information need [8]; thus, there is no related context available. Similarly, the average amount of queries per search task is less than four, based on data released by widely used search engines [4, 17]. For instance, there is an average of 3.2 queries per task [7] and 3.5 queries per task [15] in publicly available search task datasets.

3 A NEW TASK SEGMENTATION APPROACH

3.1 Task segmentation problem

Tasks are recognized as a good atomic unit to divide search logs [8, 10]. Thus, to extract semantically related patterns from a chronologically ordered search log, it is crucial to identify which queries relate to the same information need [18]. More formally, the task segmentation is defined as follows. Let us consider a sequence of queries $S = \{q_1, ..., q_n\}$, a collection of pairs of successive queries $C = \{(q_1, q_2), (q_2, q_3), ..., (q_{n-1}, q_n)\}$ and its respective known labels $L = \{l_1, ..., l_{n-1}\}$ where $l_i \in \{0, 1\}$, $\forall i \in [\![1; n-1]\!]$. The task segmentation goal is to correctly predict the labels of unseen query pairs, where $l_i = 1$ or $l_i = 0$ indicate if q_{i+1} and q_i belong to the same search task or not, respectively.

3.2 The proposed BiRNN approach

Our proposed BiRNN-based architecture (Figure 1) processes search query vector representations and learns to identify task boundaries

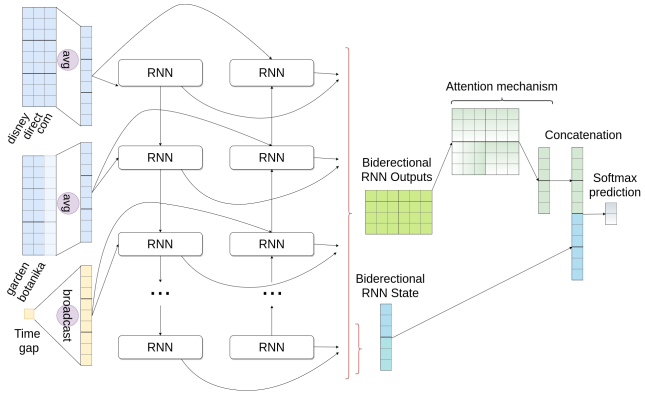

Figure 1: Segmentation architecture with a bidirectional recurrent neural layer along with an attention mechanism.

in pairs of successive queries. The architecture relies on a bidirectional recurrent layer, a commonly used layer for processing sequential information [1, 6, 9]. Furthermore, some studies in the field of Neural Machine Translation showed that intermediate output states from recurrent layers could significantly improve the performance of the initial models [11]. Therefore, following these conclusions, we include an attention mechanism in the segmentation architecture to leverage the information from the intermediate output states. Additionally, completely different queries from a lexical standpoint might represent very similar information seeking needs. For example, “constantinople" and “istanbul archeology" could represent semantically related information needs because Istanbul and Constantinople refer to the same city, but there is no lexical similarity between the two queries [7, 15]. These discrepancies could affect the ability of methods that rely on character-level information to extract task patterns from query logs. Thus, we do not use character-level information to encode queries. Instead, we use word embeddings to capture the semantic meaning of the queries. In the following, we describe the components of the proposed approach.

3.2.1 Input representation. We used word embeddings to represent queries in the search log (Figure 1). As queries are short texts, we map every query to a single vector by averaging together all the word embeddings of the query. This query representation – used in several recent studies [5, 12, 15] – allows us to keep both the syntactic and semantic information of the query content [15]. Moreover, we concatenate the time span in seconds between the queries in the pair. For the experiments to evaluate the model with adjacent queries for context, every adjacent query vector is appended to the query pair representation, respecting the sequential order in which queries appear in the search log.

3.2.2 Recurrent neural network. An RNN is a type of neural network ideally suited to process sequential information. In contrast with previous methods [10, 12, 15, 18], RNNs tend to scale adequately and perform well when dealing with sequential data [1]. RNNs store information from input sequences by using iterative function loops [14]. To store information, RNNs have a hidden state vector, which works as a memory while the network processes

the sequence in the forward direction [14]. BiRNNs were proposed to process the input sequence simultaneously in a forward and backward direction. To do so, they compute forward hidden states and backward hidden states. Concatenating forward and backward hidden states enables the generation of outputs that leverage information from preceding and following steps in the sequence [11].

In our proposed approach, we add an attention mechanism [11] at the output of the bidirectional RNN (Figure 1). The forward and backward hidden states of the bidirectional recurrent layer are concatenated to the attention mechanism output before applying dropout. The proposed BiRNN-based model uses a fully connected layer that takes the dropout result as input (Figure 1). Then, Softmax is used to generate the results in the fully connected layer, indicating if the two queries are part of the same task or not.

3.2.3 Configurations of the proposed BiRNN approach. Four alternative configurations are studied for the proposed BiRNN approach as a result of two hyperparameters: time span location and recurrent unit type. The time span d_{time_span} can be placed in the initial layer (Figure 1) or the concatenation layer [3], at the attention mechanism output. The latter is inspired by previous works such as [16], where the late concatenation of extra features improved performances in text classification. Regarding recurrent unit types, we consider both LSTMs and Gated Recurrent Units (GRUs) [2, 14].

Dataset	WSMC12		CSTE	
Segmentation model	Accuracy	F-score	Accuracy	F-score
Logistic regression	0.733	0.238	0.634	0.322
K-Nearest Neighbors	0.865	0.701	0.708	0.460
SVM, linear kernel	0.735	0.022	0.669	0.232
SVM, RBF kernel	0.742	0.055	0.671	0.105
Naive Bayes	0.773	0.306	0.643	0.223
QDA	0.323	0.428	0.534	0.553
Random Forest	0.862	0.742	0.725	0.502
AdaBoost	0.862	0.739	0.721	0.504
GPC	0.905	0.811	0.720	0.417
Decision Tree	0.882	0.777	0.759	0.541
HBSSM	0.886	0.813	0.656	0.627
BiRNN LSTM - time at AL	0.921	0.861	0.784	0.651
BiRNN GRU - time at AL	0.927	0.867	**0.789**	0.648
BiRNN LSTM	0.931	0.875	0.788	**0.663**
BiRNN GRU	**0.937**	**0.884**	0.751	0.604

Table 1: Model performance with GRUs, LSTMs, and time span at the attention layer (AL). Results are statistically significant against the baseline with $p \leq 0.05$.

4 EXPERIMENTS AND RESULTS

4.1 Datasets and evaluation considerations

Two datasets were considered in this work: the Webis Search Mission Corpus 2012 (WSMC12) dataset [7] and the Cross-Session Task Extraction (CSTE) dataset [15]. To evaluate the performance of the different approaches, we used the *Accuracy* and *F-score* measures with 10-fold cross-validation. The Student's paired t-test provided the test for statistical significance [15]. We used the GloVe publicly available pre-trained word vectors [13] for computing the search query vector representations. Word vectors have a coverage of 96.2% for the WSMC12 dataset, which has 8840 queries, and 84.5% for the CSTE dataset, which has 1424 queries. To train the segmentation architecture, we minimized the cross-entropy loss with the Adam optimizer. The learning rate was set to 10^{-4}, batch size to 256, layer size to 32, training steps to 6 x 10^4, and dropout to 0.3.

4.2 Evaluation results

4.2.1 Comparison of results. We compared the proposed approach to the state-of-the-art HBSSM approach as a baseline. We also considered other supervised machine learning approaches like Support Vector Machine (SVM) with linear kernel and Radial Basis Function (RBF) kernel (γ=2, C=1), K-Nearest Neighbors (n=3), Gaussian Process Classification (GPC) with ($\kappa[\gamma$=1]), and Quadratic Discriminant Analysis (QDA), along with CA-LSTM [3] to evaluate the proposed model in several context scenarios (Tables 1, 2). Additional to the query pair representation (Section 3.2.1), we provided query texts and clicked URLs, when available, to the baseline HBSSM approach so that it was possible to compute all the heuristics specified for the model. We also flattened the query pair representation for methods that required a unidimensional vector as input, ensuring that all methods got the query pair and its time span as input information.

Our proposed segmentation architecture obtains very satisfactory results, surpassing the remaining methods used for comparison. Moreover, GPC – the best performing method from the traditional machine learning approaches – gets results close to the HBSSM performance and matches the accuracy of CA-LSTM, even exceeding it in some context scenarios. Nonetheless, results from our proposed segmentation architecture outperform the previous recurrent model in all context scenarios. The trained BiRNN GRU architecture is also faster than the HBSSM baseline. We measured the query pair processing time using a CPU-only virtual instance with a CPU frequency of 2.397 GHz, taking the average time per query pair for all the pairs in the WSMC12 dataset. The trained BiRNN GRU architecture took around 15ms per query pair, while the HBSSM approach took around 61ms. This difference in execution time represented a speedup of 4 for our proposed architecture.

Likewise, concatenating the time spans to the attention mechanism decreases the performance of the proposed architecture. The best scenario happens when the query vectors and the time span information are part of the input sample representation. This configuration allows the computation of hidden state vectors inside the bidirectional layer that rely upon both the query pair semantic content and the time between them. Similarly, the bidirectional output vectors depend on both semantic and time information to feed the attention mechanism.

4.2.2 Impact of query context. We also analyzed how the model behaves when adding adjacent queries for session segmentation. We computed query vectors for each adjacent query and appended the query vectors to the original pair, respecting the order in which they appeared in the original search log. Half of the adjacent queries precedes the query pair, while the remaining half follows the query pair in the chronological search log. When the input includes adjacent queries for context, we do not see any improvement in the accuracy of the segmentation architecture. Although the addition

No. of queries	2	4	6	8	10
CA-LSTM	0.863	0.878	0.904	0.898	0.903
BiRNN LSTM	0.929	0.921	0.920	0.915	0.912
BiRNN GRU	**0.935**	0.920	0.927	0.919	0.917

Table 2: Segmentation model performance for the WSMC12 dataset with additional adjacent queries for recurrent architectures. Results are statistically significant with $p \leq 0.05$.

of adjacent query vectors still exceeded other baseline methods, overall, none of the results improved the performance of the query pair without context (Table 2). Such a result is essential because the average amount of adjacent queries per search task is around three. Similarly, simple information needs like fact-finding are usually solved with only one query [8]. Furthermore, forward context can be a critical drawback, especially when supporting user information needs. Applications like query suggestion must respond on the fly. They cannot wait for future queries to provide context to the model.

	No pre-training		Pre-training	
Segmentation model	Accuracy	F-score	Accuracy	F-score
BiRNN LSTM - time at AL	0.784	0.651	0.787	0.649
BiRNN GRU - time at AL	0.789	0.648	0.782	0.644
BiRNN LSTM	0.788	0.663	0.769	0.644
BiRNN GRU	0.751	0.604	**0.803**	**0.665**

Table 3: Fine-tuning the recurrent architecture to realize segmentation on the CSTE dataset.

4.2.3 Fine-tuning with a smaller dataset. Scarce data poses challenges to deep learning architectures, and fine-tuning is one of the alternatives to deal with small datasets [1]. Table 3 presents the results for the fine-tuning experiments. When performing 10-fold cross-validation on the CSTE dataset, without doing fine-tuning, we got an accuracy of 0.751. After pre-training with the WSMC12 dataset, the accuracy increased to 0.803. Thus, the BiRNN architecture configuration with the time span at the input and GRU recurrent units gave the best performance when fine-tuning. Overall, it represents the best performance with the smaller CSTE dataset when considering all the tested methods. The results mentioned above highlight the ability of the model to perform well when processing datasets with a limited number of samples, a common scenario because of the difficulty for obtaining large search query logs and the cost associated with labeling them [18].

5 CONCLUSION

We proposed a bidirectional RNN architecture with an attention mechanism for segmenting search query logs by identifying task boundaries on them. Experimental results showcase the improvement of the proposed architecture over the baseline method, outperforming the other approaches used for comparison. Once trained, the proposed model is also several times faster than the heuristics-based baseline. Furthermore, there is no need for clicked URLs, character-level representations, or additional queries surrounding the query pair to provide context to the model. This result is especially relevant given the mean number of queries per task in datasets from widely used search engines and the fact that information needs like fact-finding and other simple tasks are usually solved with only one query. Also, the proposed segmentation model performs well when fine-tuning with a smaller query log dataset. Fine-tuning performance is useful given the scarcity of publicly available labeled collections from search engines and the cost of labeling large search logs. Future work includes, first, the use of other text representations to encode queries and the analysis of its impact on model metrics. Secondly, we will also investigate attention mechanism alternative configurations and other recurrent units to modify the architecture of the segmentation model.

REFERENCES

[1] Christof Angermueller, Tanel Pärnamaa, Leopold Parts, and Oliver Stegle. 2016. Deep learning for computational biology. *Molecular systems biology* 12, 7 (2016), 878.

[2] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[3] Cong Du, Peng Shu, and Yong Li. 2018. CA-LSTM: Search Task Identification with Context Attention based LSTM. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 1101–1104.

[4] Daniel Gayo-Avello. 2009. A survey on session detection methods in query logs and a proposal for future evaluation. *Information Sciences* 179, 12 (2009), 1822–1843.

[5] Pedro Gomes, Bruno Martins, and Luís Cruz. 2019. Segmenting User Sessions in Search Engine Query Logs Leveraging Word Embeddings. In *International Conference on Theory and Practice of Digital Libraries*. Springer, 185–199.

[6] Alex Graves. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer, Berlin, Heidelberg.

[7] Matthias Hagen, Jakob Gomoll, Anna Beyer, and Benno Stein. 2013. From search session detection to search mission detection. In *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*. 85–92.

[8] Marti Hearst. 2009. *Search user interfaces*. Cambridge University Press, Cambridge, CB2 8BS, UK.

[9] Xueliang Liu. 2017. Deep Recurrent Neural Network for Protein Function Prediction from Sequence. *arXiv preprint arXiv:1701.08318* (2017).

[10] Claudio Lucchese, Salvatore Orlando, Raffaele Perego, Fabrizio Silvestri, and Gabriele Tolomei. 2011. Identifying task-based sessions in search engine query logs. In *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 277–286.

[11] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).

[12] Rishabh Mehrotra and Emine Yilmaz. 2017. Extracting hierarchies of search tasks & subtasks via a bayesian nonparametric approach. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 285–294.

[13] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[14] Anand Rajaraman and Jeffrey David Ullman. 2011. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA.

[15] Procheta Sen, Debasis Ganguly, and Gareth Jones. 2018. Tempo-lexical context driven word embedding for cross-session search task extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 283–292.

[16] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. ACM, 373–382.

[17] Michael Völske, Ehsan Fatehifar, Benno Stein, and Matthias Hagen. 2019. Query-Task Mapping. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 969–972.

[18] Hongning Wang, Yang Song, Ming-Wei Chang, Xiaodong He, Ryen W White, and Wei Chu. 2013. Learning to extract cross-session search tasks. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1353–1364.