ECAI 2008
M. Ghallab et al. (Eds.)
IOS Press, 2008

doi:10.3233/978-1-58603-891-5-827

ContextAggregator: A heuristic-based approach for automated feature construction and selection

Robert Lokaiczyk and **Manuel Goertz**[1]

Abstract. Our research goal is to work towards a personal context-aware assistance and retrieval of relevant resources to computer users during a certain work task. This paper presents a general-purpose, algorithmic approach for automated context aggregation by heuristic-based feature construction. Our implementation of the context reasoning layer combines lower-level context features to new aggregated higher-level context features. Our approach allows – in contrast to most other approaches – an automated feature combination to achieve a high prediction accuracy of the user's work task.

Introduction

Recent work in personal information and knowledge management systems often focuses on context awareness [5] and task orientation [3, 9]. Which, given a determined work task, provide the user with suitable learning resources relevant for the current learning need in the current work task. A crucial factor for fulfilling the vision of in-place and in-time e-learning systems is the user's context. Taking the user's current task into account the systems are able to provide adaptive assistance and learning resources. Forms of workplace-integrated learning support might be displaying a list of task-relevant documents in an enterprise environment. In [7] and [3] resources are determined by querying the (pre-modelled) semantic network given a description of the current work task. Consequently, our goal should be to determine the current work task of the user automatically only by means of available context information on the desktop and not by manual input by the user.

1 Context

We focus on knowledge-intensive work on the desktop of the computer worker. Therefore, we define *Desktop Context* – in accordance with [1] – as all measurable environmental settings that surround the user desktop work. Technically, these settings are monitored by desktop context sensors that collect system events and user interaction with the workbench. The context sensors are implemented as software hooks that operate on operation system level and log the data continuously. Thereby, the layer of context elicitation is completely transparent and unobtrusive to the user. The collected context events are encoded in a data stream which can be used as a feature stream for further processing.

Whole tasks can be seen as slices of the event stream consisting of typical events correlating with a certain work task. The sequence of context events reflects the users actions during the work process. Context events include keystrokes, application launches, full-text of documents etc. Based on the user's context information it is possible to predict the user's work task.

[1] SAP Research, Darmstadt, Germany, email: firstname.lastname@sap.com

2 Approach

The basic approach of understanding the problem of task detection as a machine learning classification is shown in [6]. Consequently we only briefly summarize the key idea. First, a reasonable amount of training data is acquired by manual annotation of the work task by user right during is work process. The user selects from a limited set of tasks which are pre-modeled and typical for the work process within the involved organization. The selected task is annotated to the collected training material of work streams recorded with the context monitor. The task prediction algorithm based on the learned model automatically classifies the active tasks using continuously recorded event streams. Whenever the classifier detects a change in the user's work tasks, a new retrieval of task-relevant resources is triggered and our personal information assistant displays a new list of associated learning resources.

2.1 ContextAggregator Algorithm

The paper presents the idea of unsupervised context aggregation. Until now most approaches of aggregating desktop events to more complex, meaningful units are manually handled by the user or previously modeled by domain experts. We differ by providing an unsupervised algorithm for context aggregation that takes the user out of the loop and is not dependent of domain-specific knowledge. The fundamental idea is combining desktop events to new events that potentially are more valuable features for work task prediction. Thereby the mutual correlation between features is taken into account to increase information gain for the prediction.

2.2 Aggregation Functions

The idea of the aggregation functions here is basically building predicates on new combinations of features that are considered as potentially more valuable features. As measurement for the impact we use information gain [8], a common feature relevance measure from the data mining area. We propose a algorithm and a set of combination functions that appears to be very prospering for our particular context aggregation problem. For combining features we use an extensive set of functions that map a number of features (n) to a new feature (see equation 1).

$$f_i : F^n \rightarrow F \tag{1}$$

For our experiments the used set of functions turns out to deliver already good results. But the extensibility of the algorithm with more

specific aggregation functions is definitely an advantage in order to receive even better results with domain-specific mapping functions.

2.3 Heuristic

For reducing both the computational complexity and the memory requirements we apply some heuristic rules that prefer certain feature combinations and reject others.

In particular we use the following heuristics to reduce the complexity of context aggregation:

I) Filter ill-defined mappings of events. As an example we can consider the function $max(date, windowname)$, which is not defined.

II) Keep statistics of transformation functions that usually lead to increased information gain. Thus, the algorithm can prefer rules that are already known to improve the result on the particular domain.

III) Skip feature duplicates. We avoid those features by checking for duplicates within the already existing feature vectors. As an example you can consider $max(max(event))$ which always reduces to $max(event)$.

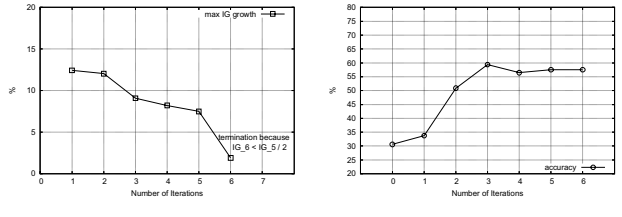
IV) Limit the stored feature set to a small subset of possible features. We keep only the topmost n features (ranked by information gain).

V) Skip feature combinations with low impact. For a potential improvement the information gain of the feature combination should be at least above the maximum of the information gain of the involved features.

With this set of rules the algorithm is quickly able to determine the most valuable feature combinations and will not take unimportant combinations into account.

3 Analysis of the Algorithms

First, the we analyse the convergence of the ContextAggregator-algorithm. As shown in Figure 1(a), the ContextAggregator-algorithm usually converges very fast, only after a few iterations. In our experiments there was no more strong improvement after about 5 iterations.



(a) Convergence of the Maximum Information Gain Growth

(b) Boosted Task Prediction Accuracy

Figure 1. Evaluation of the Proposed Algorithm

For evaluation purposes we (see Chapter 1) collected context data together with annotated task labels during a work process (14 unique users; 18 work hours). To measure the improvement of aggregating the context of the collected training material, we apply an n-fold cross-validation where n is the number of distinct users. We calculate the averaged performance metrics from the individual data segmentations. The separation of training data for each user is necessary in order to really prove that the learned knowledge from the training data is really transferable to the separated user whose own training material is not in the particular training set. As classification algorithm we use Naive Bayes, since it has the theoretical minimum error rate in comparison to all other classifiers [4] and practical experiments indicate a good accuracy even if the independence assumption is violated [2].

In order to prove the boosted accuracy with automatically derived higher-level context information we compare the accuracy values of the prediction algorithm with context aggregation to those without. Obviously, the context aggregation yields to an increase in prediction accuracy which can be seen in Figure 1(b). This result is significant at a confidence of 99%.

4 Summary

In this paper we propose a multi-purpose context aggregation algorithm based on heuristic rules that is able to construct more relevant features out of the large number of possible context events. Furthermore, we evaluate the algorithm on the data of a user study for the purpose of user task prediction and show a significant improvement over the basic non-aggregated version. By using a number of simple heuristics we are able to reduce the computational complexity and memory requirements of the aggregation algorithm.

REFERENCES

[1] Anind K. Dey. Understanding and using context, 2001.

[2] Pedro Domingos and Michael J. Pazzani, 'Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier', in *International Conference on Machine Learning*, pp. 105–112, (1996).

[3] Olaf Grebner, Uwe V. Riss, Ernie Ong, Marko Brunzel, Thomas Roth-Berghofer, and Ansgar Bernardi. Task management for the nepomuk social semantic desktop (poster). 4th Conference on Professional Knowledge Management - Experiences and Visions -, March 2007.

[4] Jiawei Han and Micheline Kamber, *Data Mining. Concepts and Techniques.*, Morgan Kaufmann Publishers, 2001.

[5] Angela Kessell and Christopher Chan, 'Castaway: a context-aware task management system', in *CHI '06: CHI '06 extended abstracts on Human factors in computing systems*, pp. 941–946, New York, NY, USA, (2006). ACM.

[6] Robert Lokaiczyk, Andreas Faatz, Arne Beckhaus, and Manuel Görtz, 'Enhancing just-in-time e-learning through machine learning on desktop context sensors', in *CONTEXT*, eds., Boicho N. Kokinov, Daniel C. Richardson, Thomas Roth-Berghofer, and Laure Vieu, volume 4635 of *Lecture Notes in Computer Science*, pp. 330–341. Springer, (August 2007).

[7] H. Mayer, W. Haas, G. Thallinger, S. Lindstaedt, and K. Tochtermann. APOSDLE - Advanced Process-oriented Self-directed Learning Environment. Poster Presented on the 2nd European Workshop on the Integration of Knowledge, Semantic and Digital Media Technologies, 30 November - 01 December 2005.

[8] Thomas M. Mitchell, *Machine Learning*, McGraw-Hill Higher Education, 1997.

[9] Jianqiang Shen, Lida Li, Thomas G. Dietterich, and Jonathan L. Herlocker, 'A hybrid learning system for recognizing user tasks from desktop activities and email messages', in *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces*, pp. 86–92, New York, NY, USA, (2006). ACM Press.