Task Instance Classification via Graph Kernels

Mark Kröll, Andreas S. Rath, Nicolas Weber, Stefanie Lindstaedt, and Michael Granitzer

Know-Center GmbH, Inffeldgasse 21a/II, A-8010 Graz, Austria
mkroell, arath, nweber, slind, mgrani@know-center.at

Knowledge-intensive work plays an increasingly important role in organisations of all types. Knowledge workers contribute their effort to achieve a common purpose; they are part of (business) processes. Workflow Management Systems support them during their daily work, featuring guidance and providing intelligent resource delivery. However, the emergence of richly structured, heterogeneous datasets requires a reassessment of existing mining techniques which do not take possible relations between individual instances into account. Neglecting these relations might lead to inappropriate conclusions about the data. In order to uphold the support quality of knowledge workers, the application of mining methods, that consider *structure* information rather than content information, is necessary. In the scope of the research project DYONIPOS, user interaction patterns, e.g., relations between users, resources and tasks, are mapped in the form of graphs. We utilize graph kernels to exploit structural information and apply Support Vector Machines to classify task instances to task models.

1 Motivation

Workflow Management Systems (WFMS) have become quite popular in organizations, because they promise to solve the problems arising from complex organizational processes. This significant contribution of WFMS to increase the productivity is generally accepted [11], but WFMS still impose too many restrictions on knowledge workers [14]. Knowledge workers are guided by goals instead of tasks and prefer significant freedom in structuring their own activities. The DYONIPOS research project[13] strives to slacken the constraints by admitting more flexibility in the work process. This additional freedom bears the risk of loosing the guiding capabilities of WFMS. Monitoring user-desktop interactions and thus capturing the knowledge worker's context [10] is one possibility to compensate this loss. The extracted information can be used to guide the knowledge worker through the work process and to intelligently deliver useful resources. However, to provide guidance in the form of propositions of the next step or of tutorials to read, the current process instance the user resides in has to be determined. Considering only information based on the content of documents that have been read or written lately might lead to ambiguous results, since process instances of a common process[1] show great variations regarding their context.

[1] The process "Write a paper" possesses many derivations, e.g., concerning different research areas as computer science, medical science and so on.

Hence, structural information is sometimes the only hint for categorizing task instances to task models and process instances to process models. In our research project graph kernels in combination with Support Vector Machines(SVMs) are applied to task instance classification to exploit the structural information in the data.
This extended abstract is structured as follows: Section 2 discusses related work in this area, Section 3 describes the information gathering, the representation of the data and the application of graph kernels for the classification task. The paper closes with concluding remarks, acknowlegements, and the list of references.

2 Related Work

In the area of task classification, Shen et al. [12] use a classifier to predict the current task based on features, e.g., window title and the file pathname, extracted from the window in focus. In [8], task assignment is based on relations of the windows on the user's desktop. Approaches including graph kernels have not been investiged yet to the best knowledge of the authors.
Relations between entities provide valuable information for mining tasks. The relatively nascent research areas *Link Mining* [3] and *Graph Mining* [16] consider relations within the dataset and attempt to exploit the topological view of structured data. Haussler [4] introduced convolution kernels, a general framework for handling discrete data structures by kernel methods. Kashima et al. [6] concentrated on the construction of graph kernels. Their graph kernel performs a random walk on the vertex product graph of two graphs.
Many areas of knowledge such as computational biology [15], computational chemistry [7], [9] and computer vision have recently been using kernel methods and graph kernels in order to learn from structured data.

3 Structural Categorization of Task Instances

This section describes the gathering of the data, how the collected data is processed, i.e., which components flow into the graph representation, and how graph kernels in combination with SVMs are utilized for task instance classification.
User interactions with the system and reactions from the system to the users interactions are monitored and stored in a RDF repository. These interactions are interpreted as events. Events are user inputs, such as mouse movements, mouse clicks, starting a program, creating a folder, a web search, or opening a file. A sequence of events are grouped together into blocks of events by using predefined static rules, which map a set of events to an event block. An example of an event block is "editing document x on page y". In this event block all events that occur from key strokes in a text editing application are aggregated and form the event block. A task instance, i.e., the execution of a predefined task model, is formed by combining event blocks that are related. Grouping together event blocks into tasks is done by calculating similarities based on common content, e.g. the bag

of words approach. However, the assignment of task instances to predefined task models cannot be based solely on content similarity. The content of two task instances belonging to the same task can vary arbitrarily, e.g., think of writing a paper in different scientific fields. Thus, a classification of task instances needs to resort to structural information, i.e., which types of event blocks occur, which resources have been used or which persons have been contacted.
The structural information of a task instance is represented as a graph. The temporal sequence of event blocks forms the basis of each graph. The used application and the corresponding user behaviour, i.e., reading or writing, define the type of an event block. Additional nodes such as resources or persons involved are attached to corresponding event block nodes of the graph. Based on that information an adjacency matrix of a task instance can be established.
The assignment of a task instance to a predefined task model is accomplished by SVMs which have been proven to be quite effective in the area of supervised learning. Using SVMs allows the definition of the algorithm's view on the data by selecting the kernel. Hence, a self-calculated (graph) kernel can be plugged into a kernel machine without needing to adapt the algorithm's procedure. We are using the LibSvm implementation [1] which allows the handing over of a precomputed kernel.
The graph kernel is computed in a similar way as described in [2]. The adjacency matrices of two task instances are transformed into a direct product graph [5]. The definition of a graph kernel in use reads as follows:
Let G_1, G_2 be two graphs, $E_\times$ denotes the adjacency matrix of their direct product $E_\times = E(G_1 \times G_2)$, and $V_\times$ denotes the vertex set of the direct product $V_\times = V(G_1 \times G_2)$. With a sequence of weights $\lambda = \lambda_0, \lambda_1, \ldots (\lambda_i \in \mathcal{R}; \lambda_i \geq 0$ for all i $\in \mathcal{N}$ the product graph kernel is defined as $k_\times(G_1, G_2) = \sum_{i,j=1}^{|V_\times|} \left[\sum_{n=0}^{\infty} \lambda_i E_\times^n\right]_{ij}$, if the limit exists.
To evaluate the graph kernel, the limit of the above matrix power series has to be calculated. The limit of series like $\lim_{n\to\infty} \sum_{i=0}^{n} \lambda_i E^i$ can be computed by means of the limit of eigenvalues of the matrix E as shown in [2]. First experiments with small sized graphs have proven to be promising. The structure similarity, that is independent of the content, is captured and used to classify the occurring task instances. Taking into account the structural information clearly underpins the advantage of using a graph kernel over a linear kernel. The linear kernel does contain the same features but lacks mapping the relations among the features.

4 Concluding Remarks

In this extended abstract we presented a novel approach for categorizing task instances to predefined tasks. The classification is based on structural information rather than content information. A task instance and its inner relations are represented in the form of a graph. The similarity measure is defined by the application of graph kernels. We are going to obtain larger, annotated datasets by monitoring user-desktop interactions of employees in the Austrian Ministry

of Finance. Having real-world datasets at hand, further experiments will be conducted.

5 Acknowledgements

The project results have been developed in the DYONIPOS project (www. dyonipos.at). DYONIPOS is financed by the Austrian Research Promotion Agency (www.ffg.at) under the project contract number 810804/9338.

The Know-Center is funded by the Austrian Competence Center program Kplus under the auspices of the Austrian Ministry of Transport, Innovation and Technology (http://www.ffg.at), by the State of Styria and by the City of Graz.

References

1. C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.
2. T. Gärtner, K. Driessens, and J. Ramon. Graph kernels and Gaussian processes for relational reinforcement learning. volume 2835, pages 146–163, 2003.
3. L. Getoor. Link mining: a new data mining challenge. *SIGKDD Explor. Newsl.*, 5(1):84–89, 2003.
4. D. Haussler. Convolution kernels on discrete structures. Technical report, UC Santa Cruz, UCS-CRL-99-10, 1999.
5. K. S. Imrich, W. *Product Graphs: Structure and Recognition.* John Wiley, 2000.
6. H. Kashima and A. Inokuchi. Kernels for graph classification. ICDM Workshop on Active Mining 2002, 2002.
7. P. Mahé, N. Ueda, T. Akutsu, J.-L. Perret, and J.-P. Vert. Graph kernels for molecular structure-activity relationship analysis with support vector machines. *J. Chem. Inf. Model.*, 45(4):939–51, 2005.
8. N. Oliver, G. Smith, C. Thakkar, and A. C. Surendran. Swish: semantic analysis of window titles and switching history. In *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces*, 2006.
9. L. Ralaivola, S. J. Swamidass, H. Saigo, and P. Baldi. Graph kernels for chemical informatics. *Neural Netw.*, 18(8):1093–1110, 2005.
10. A. Rath, M. Kröll, S. Lindstaedt, and M. Granitzer. Low-level event relationship discovery for knowledge work support. 2007. ProKW2007 Productive Knowledge Work : Management and Technological Challenges.
11. U. V. Riss. Knowledge, action, and context: A process view on knowledge management. In *Wissensmanagement*, pages 555–558, 2005.
12. J. Shen, L. Li, and T. Dietterich. Real-time detection of task switches of desktop users. In *Proceedings of the IJCAI. Hyderabad, India. pp. 2868-2873*, 2007.
13. K. Tochtermann, D. Reisinger, M. Granitzer, and S. Lindstaedt. Integrating ad hoc processes and standard processes in public administrations. In *Proceedings of the OCG eGovernment Conference, Linz (Austria)*, 2006.
14. W. van der Aalst and M. Weske. Case handling: A new paradigm for business process support. *Data Knowl. Eng.*, 53(2):129–162, 2005.
15. J.-P. Vert. Kernel methods in genomics and computational biology. Technical report, HAL:ccsd-00012124, 2005.
16. T. Washio and H. Motoda. State of the art of graph-based data mining. *SIGKDD Explor. Newsl.*, 5(1):59–68, 2003.