Real-time task recognition based on knowledge workers' computer activities

Saskia Koldijk
TNO & Radboud University Nijmegen
The Netherlands
s.koldijk@cs.ru.nl

Mark van Staalduinen
TNO
The Netherlands
mark.vanstaalduinen@tno.nl

Mark Neerincx
TNO & Technical University Delft
The Netherlands
mark.neerincx@tno.nl

Wessel Kraaij
TNO & Radboud University Nijmegen
The Netherlands
wessel.kraaij@tno.nl

ABSTRACT

Motivation – Supporting knowledge workers in their self-management by providing them overviews of performed tasks.

Research approach – Computer interaction data of knowledge workers was logged during their work. For each user different classifiers were trained and compared on their performance on recognizing 12 specified tasks.

Findings/Design – After only a few hours of training data reasonable classification accuracy can be achieved. There was not one classifier that suited all users best.

Take away message – Task recognition based on knowledge workers' computer activities is feasible with little training, although personalization is an important issue.

Keywords

Task recognition, field study, unobtrusive sensing, pattern recognition, personalization.

INTRODUCTION

Nowadays, many people spend their working days at a computer, coordinating different activities in several projects to create information products. We refer to these people as knowledge workers. Typically, they have to self-manage their work to accomplish all their tasks. Their course of action is not always self-planned but also determined by external causes, like phone calls, mails, information requests, other persons or appointments (Czerwinski, Horvitz, & Wilhite, 2004), which easily results in a fragmented way of working. So, a good overview of tasks is important for them, but rather difficult to maintain. The goal of our research is to support knowledge workers with tools. This paper aims at automatic task recognition to provide overviews of tasks performed.

Knowledge workers rely on software for communication, information gathering, document creation and work planning, so a vast collection of digital traces is left behind on their computer. These are available in the form of mouse motion, click events, key presses and active window changes. We use these traces to automatically infer what task a user is currently performing. In this way we automatically create a real-time overview of tasks for the user in an unobtrusive way.

As research has shown, more awareness of one's own working process can have beneficial effects on the on-task behaviour and adherence to scheduled activities (Richman, Riordan, Reiss, Pyles, & Bailey, 1988). A study by Johnson and White (1971) showed that mere self-observation caused a positive change in behaviour. By being able to easily look back at their behaviour, knowledge workers might get a better grip on their work style and improve it. Cognitive load and stress might be decreased.

Some systems that provide overviews of computer activity exist (e.g. Slife[1], RescueTime[2]), but they present low-level data in the form of time spent per application and websites browsed. They require the user to interpret for which task a specific program or website was used. In our research, minimal effort should be required from the user. So we aim at automatic recognition of tasks based on computer activities. We use not only application information, but also typical patterns of behaviour that originate from mouse and keyboard.

In the field of activity recognition, various activities are automatically recognized, for example activities in an adventure game (Albrecht, Zukerman, Nicholson, & Bud, 1997) or computer activities, like filling in a form or planning a meeting (Rath, Devaurs, & Lindstaedt, 2009). These activities have rather clear structures, involving predefined steps (see Natarajan, Bui, Tadepalli, Kersting, and Wong (2008)). Therefore, often model-based classification is applied, with logical models assuming a plan library (e.g. Goldman, Geib, & Miller, 1999) or Markov models, modelling the sequence of actions in time (e.g. Albrecht et al., 1997). Moreover, most models are applied to simple problems

[1] http://www.slifeweb.com

[2] http://www.rescuetime.com

in a controlled environment. Our models will be tested in a field study. The recognition of knowledge workers' tasks on the basis of computer activities is a new domain with different characteristics, where tasks are less structured and task sequences are more spontaneous. Whether task recognition in this domain is feasible is thus a challenging research question.

This paper is organized as follows. First, we describe our framework for task recognition, then we explain how we evaluated this framework in a field study. Thereafter, our analyses and results are presented, followed by a discussion and conclusions.

TASK RECOGNITION FRAMEWORK

To recognize knowledge workers' tasks automatically, a framework is necessary that specifies the mapping from low level computer interaction data to performed tasks. The following components are required to realize this framework:

- A set of task labels that users intuitively use.
- A number of useful features obtained from computer interaction data.
- Different classifiers that map low level activity features to the defined task labels.

These components are described in the next three subsections.

Task Labels

To obtain more knowledge about tasks that knowledge workers typically perform, and which task labels they intuitively use, we developed a questionnaire. In total 47 employees from TNO (Netherlands Organization for Applied Scientific Research) with various backgrounds and different functions completed this online questionnaire.

The answers to the questions 'What tasks do you perform and how do you use your computer to realize this task?' and 'Describe a typical working day' were manually grouped into sets of similar answers. Task categories that clearly arose from the data were email, meeting and planning. These were mentioned by nearly anyone. Depending on the specific role or expertise of the knowledge worker several project tasks were mentioned, like searching for information, analysing data, making a presentation or writing a report. Many people also listed phoning, traveling, using social media, coffee breaks, talking with colleagues, doing some private Internet browsing, or having lunch.

The appropriateness of our identified task labels was confirmed by several knowledge workers. We investigated automatic task recognition for those tasks that are performed using a computer:

- Read mail
- Write mail
- Organize/ archive data
- Plan
- Program
- Write report/ paper
- Search information
- Read article/ text
- Make presentation
- Make overview
- Create visualization
- Analyse data

We learned that knowledge workers do not intuitively think in terms of applications to categorize their activities. They have a specific purpose or task in mind, which often requires the use of several applications. The tasks are in focus and the applications used depend on these tasks. Important to note is that some applications, like PowerPoint, are used for different tasks. Therefore task recognition is not a simple one-to-one mapping between an application and a task. Users also switch between different applications while executing one task, which became clear from the descriptions of some respondents. Our recognition model should be robust to this behaviour.

Features

Automatic task recognition requires relevant features. In our research, computer interaction data is used, which should be automatically logged. From this raw data useful features should be extracted, such that the classifier can discriminate between tasks.

We used uLog (software developed by Noldus Information Technology[3]) to log mouse and keyboard actions, as well as the applications used. Thereafter, this raw data was processed to extract relevant features. All these features were calculated for a 5 minute time segment, which we assume to be long enough to average out fluctuations, but fine grained enough not to lose useful information. In this way we calculated for example how often the user clicked within the 5 minute segment, or how much of the time a certain application was in focus within this 5 minute segment.

Mouse features include

- the number of clicks and scrolls within the time frame.

Keyboard features include

- the amount of characters and special keys typed,
- the number of spaces and backspaces.

Application features include

- the application that was mainly in focus during the five minute time frame,
- features for typical applications like Word or Outlook, which indicate what percentage of time these applications were in focus.

Other features used are

- the number of different applications used within the time frame,
- the number of switches between applications,
- the time of the day.

[3] http://www.noldus.com

Classifiers

For mapping simple features to higher level tasks, a classifier is used. All features determined for one time segment are provided to a classifier, which assigns a task label to this time segment. As knowledge workers' tasks do not have a clear predefined structure, which could be modelled, we chose to use several common and rather simple data-based classifiers:

- KStar
- Naïve Bayes
- Decision Tree
- Multilayered Perceptron

For all classifiers we used Weka (Hall et al., 2009) with default settings. To investigate which of the different classification principles is most suitable in our domain, we compared the performance and learning curves of these classifiers.

The reason to use a single time segment for classification is that it simplifies the model, which yields fast task recognition and requires a small number of parameters to be estimated. This seemed a good starting point to us. This model is easier to train, than more complex temporal models, where the label of a segment is also determined by information from previous time segments. Moreover, training a temporal model requires more ground truth labels than our model, and in a real-world setting such a large labelled dataset is difficult to acquire.

APPROACH FOR FRAMEWORK EVALUATION

To evaluate our task recognition framework, we performed a field study in which data was collected from knowledge workers who were performing their daily job. These workers regularly annotated which task they were performing. This annotated data set was then used for several analyses. We aimed to investigate how good our framework is, in terms of classification performance and learning speed, for recognizing tasks performed by different knowledge workers.

We now explain the tool to collect annotated data in a user friendly way and then describe the method of our user study.

Tool For Collecting Annotated Data

For our study, the participants had to annotate their activities with task labels while working at the computer. A simple pop up reminding them to indicate which task they were currently performing was perceived as very annoying. Therefore, we created a more user friendly data annotation tool, which makes the labelling easier by suggesting task labels to the user. Classifiers were trained on the initially collected dataset of our pilot study. These are then used to automatically classify the previous five minutes of user activity. The recognized task label of one of the classifier types is then presented to the user in a small pop-up. The user can look back at the suggested task labels of the previous hour and confirm or correct them (see Figure 1). This approach makes it easy to check or correct activity labels whenever the user wished to. After one hour of new data the classifier is retrained to optimally predict suitable task labels for this user.

Besides making labelling of activities easier, we added two types of visualizations to make the use of the program more interesting for the participants. The first visualization depicts the performed tasks as a pie chart (see Figure 2). This gives the knowledge workers the possibility to look back and see which kind of tasks they were mainly performing over the days. The second visualization shows the activities of the knowledge workers as a Gantt chart (see Figure 3). In this visualization they can easily see the course of activities over the day. Our idea was that presenting users these visualizations gives them insights in their way of working and makes it more important to them to correctly label their activities.

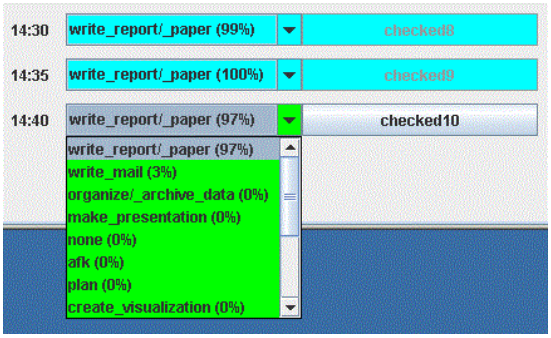

Figure 1: View to check or correct the automatic labelling.

Method

The exact method we followed for collecting annotated user data is described in this section.

Participants

Eleven knowledge workers employed at TNO volunteered to participate in our two week data collection period (10 male, 1 female). All participants typically spent most of their working day at the computer and carried out a diverse set of typical knowledge worker tasks.

Materials

The participants worked at their regular work place on their own Windows desktop computer with mouse and keyboard. The logging tool uLog was installed on the machines to capture mouse, keyboard and application activity. The logging files were read out by a Java program and stored in a triple store database (Jena) on a server for further access. Another Java program was used to fetch the current activity data from the database (using SPARQL) and apply various classifiers from the Weka machine learning toolkit in order to suggest a task label to the user.

Procedure

First of all the required software was installed on the participants' computers and its usage was explained shortly. The knowledge workers were instructed to start

up the software at the beginning of the day and work as usual. During their work, the data capturing programs ran without attracting attention. Every five minutes, the recognition program analysed the user's activity data and suggested a task label to the user in a small pop-up window. All participants used this same setup. They were told to regularly check the suggested task labels and correct them when necessary, either immediately after the pop-up or within one hour via the dashboard view (see Figure 1). It was explained to the participants that they could access some simple visualizations of the activities of the days, which were automatically made, via the dashboard whenever they wished to.

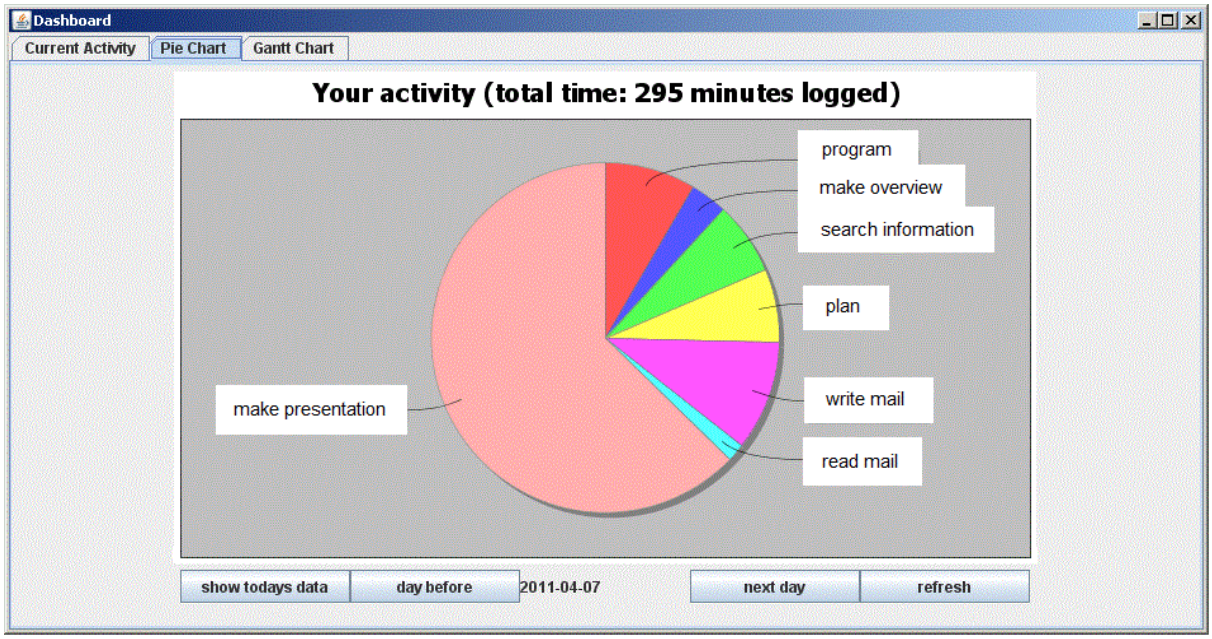

Figure 2: Dashboard with Pie chart visualization showing amount of spent time per task.

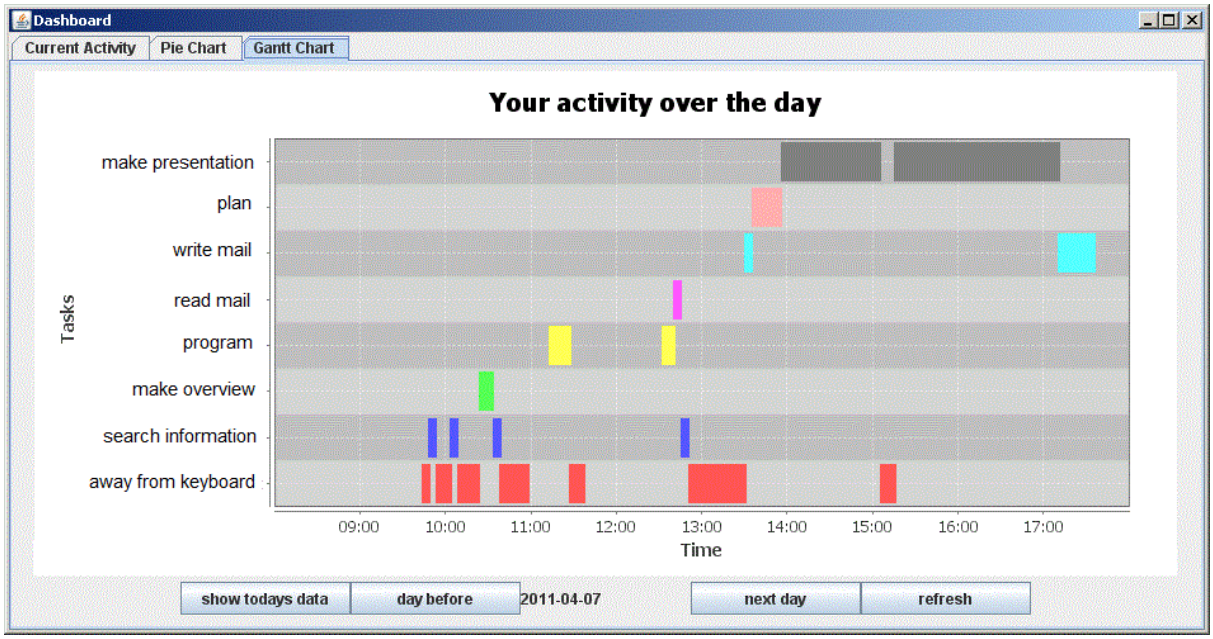

Figure 3: Dashboard with Gantt chart visualization presenting tasks performed during the day.

ANALYSES AND RESULTS

The annotated data sets resulting from our field study were used for several analyses. In this section we present the analyses performed and the results obtained, beginning with a check on our chosen task labels and features. In the next subsection, the comparison of different classifiers will be described. Finally specific analyses regarding individual differences between users are presented. For results in full detail see Koldijk (2011).

The data collection phase resulted in eleven datasets, one for each participant. For a reliable ground truth only data with labels explicitly checked by the user were used in our analyses. In table 1 the amount of checked labels per user can be seen. As user J and B checked too little labels, their data was excluded from further analyses.

Table 1: Dataset - amount of checked labels per user. (Users ordered on amount of data, users J and B were excluded from further analysis because of too little data)

User	A	C	K	I	E	G	H	D	F	J	B
# labels	522	156	144	108	72	42	36	36	30	6	3
in hours	43.5	13	12	9	6	3.5	3	3	2.5	0.5	0.25

Task Labels and Features

First of all, we tested whether the defined task labels and the chosen set of features were suitable. Only the main insights are presented here (for more details see Koldijk, 2011).

Regarding the task labels we considered confusion matrixes. In general, our task labels seemed appropriate. Typical confusions of tasks were mainly due to some tasks involving other tasks as subtasks (e.g. searching information being part of writing a document).

Regarding our chosen features, we analysed their information gain. All our features turned out to be useful. Information about applications turned out to be a good feature among users, whereas mouse and keyboard activity as well as work style (e.g. switching behaviour) are good features on a per user basis.

Comparison of Classifiers

Next, we compared the selected classifiers in terms of performance and leaning speed. Details about the analyses and results are presented in the following two subsections.

Performance

We used the Weka machine learning toolkit to train and test several classifiers, in order to answer the question which classifier is best in recognizing tasks. The performance of the classifiers was measured as percentage correctly classified instances. For performance evaluation we applied 10 fold cross-validation. To make the estimate more reliable we ran this whole process ten times and averaged the results over the runs.

Labelling each segment simply as the majority task with Weka's ZeroR classifier yielded us a baseline accuracy. We compared the performance of the following classifiers: KStar, Decision Tree, Naive Bayes and Multilayered Perceptron. All labelled data of one user at a time was used to train and test a classifier. This was repeated with all nine users' data sets.

As you can see in Figure 4, for each user all tested classifiers performed better than baseline (which was given by ZeroR). It differed per user which classifier achieved the best performance. For example, you see that the Perceptron was clearly best for user A with a final classification accuracy of about 70%, whereas for user I Naive Bayes gave best results with 80% accuracy. For user E KStar slightly won with 75% accuracy.

From our analysis we can conclude that the classification accuracy is reasonably high in this office setting, but it is impossible to say which of the classifiers generally achieves the best performance.

The different classifiers use very different principles to discriminate between tasks. There is thus not one principle that clearly works best in this domain. It might depend on the specific work style or characteristics of the user which method is most suitable. We analyse the differences between users in more detail in the section on individual differences.

Learning Curves

As a next step we investigated which classifier is fastest in learning to classify tasks. We simulated the growths of the data set in order to analyse the learning process of the classifiers. The user's complete data set was first of all split into 10 folds, one of these folds held apart for testing. From the remaining folds data was randomly sampled creating increasingly large training portions.[4] The first training portion contained 3 sampled data instances, the next 6, 9 and 12 instances. From then on the training portion size grew with 6 instances (= half an hour of data). Every classifier was then trained on each of these training portions, always using the fixed test sets to evaluate their performance. We plotted the classifier performances for different data set sizes as learning curves (values again averaged over 10 test folds and 10 runs).

Figure 4 plots the learning curves per user. It shows that, in general, the performance of the classifiers was at 80% of its maximum after only about 30 instances, which is only 2.5 hours of training data. The particular form of the learning curves differed per user. For user K, all classifiers learned slowly, whereas for user E they all learned quickly. For user I, there was a great

[4] Note that by sampling, each training portion has about the same task label distribution as in the total set. The instances were not added in the order they actually appeared during data collection, as in that case the points in the learning curves would have been very dependent on the specific task mix performed at that time.

difference between learning curves, with Naive Bayes quickly achieving a high performance and KStar performing badly, whereas for user K, all curves were mingled up, showing no clear winner in terms of learning speed.

From this analysis we can conclude that the classification is in general learned quickly in this setting, but it is impossible to say which of the classifiers generally learns quickest. Again, specific characteristics of the users seem to influence how fast a model is learned and which classifier is most suitable.

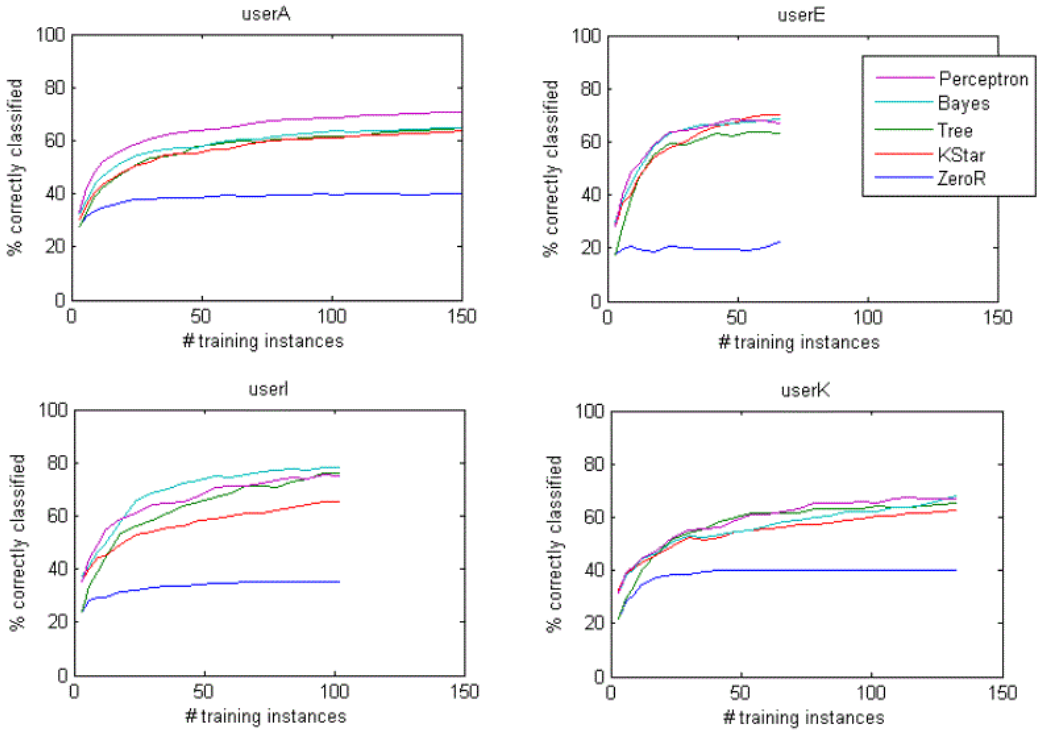

Figure 4: Learning curves for the different classifiers, for some selected users. Note: ZeroR provides a baseline.

Individual Differences

We saw great variance in both final performance of the classifiers and their learning speed between users. This poses the following questions:

- Where do these performance differences come from, i.e. how do the users differ?
- Given these individual differences, how does a trained model perform on a new user?

Differences Between Users

A first aspect we considered are the differences in the users' tasks. The distribution of tasks that the knowledge workers performed during the data collection period was analysed. Our results show that different users performed a different task mix. Some task combinations may be better distinguishable than others, so this can explain differences in classification performance.

A second aspect we considered is the typical pattern of behaviour of the users. Therefore we analysed the distribution of clicks, typing or other features per user and task. It turned out that even when users were performing the same task their behaviour differed (Koldijk, van Staalduinen, Raaijmakers, van Rooij, & Kraaij, 2011). For example user G typed extraordinary many characters when writing a report and in general clicked more often than other users. Statistical analysis in form of a 12 (tasks) x 9 (users) MANOVA with all features as dependent variables showed a significant effect of task and user on almost all features. This means not only the task, but also the specific users are distinguishable on basis of the measured behaviour.

These results hint at different users having a different way of working. They might for example differ in work style, for example thinking a lot and typing a sentence in one go versus quickly typing and retyping things. Or they might differ in mouse use, for example using mainly the keyboard to navigate versus using the mouse to point and click. These individual characteristics also make task recognition more or less easy to learn for various classifiers.

From these analyses we can conclude that the task mix of the users and their typical behaviour is very individual. This explains why there is no ‘one classifier suits all' solution.

Generalizability of the Classifiers

Analyses thus far indicate that task recognition is very personal. It is thus the question whether a classifier can be trained on a set of user data and effectively be used to classify a new user's behaviour .

To answer this question we first trained a classifier on the data of user A. We used this trained classifier to classify the test sets of other all users. Our results show that although the trained classifier worked fine on user A's test set it reached a performance of only 20% on average on other users test sets. We can conclude from this that a classifier trained on one user does not work on other users data.

Then we tested whether a classifier could become more robust in classifying a new user when it was trained on a mix of several users' data. The idea was that the classifier would not model specific details of one user, but pick up general patterns common among users. We created train sets by sampling 30 instances per user of all but one user and trained classifiers upon these data sets. Then we tested its performance on the left out user's data to test the generalizability of the model.

It turned out that the average classification performance was only 20 to 30% in this setting. This is better compared to training on one user's data, but far from satisfactory. We can conclude from this analysis that also a classifier trained on a mix of users' data does not generalize well to new users.

DISCUSSION

Our research showed that task recognition in the domain of real-world knowledge worker activities is possible, but there is no clear recommendation to which type of classifier to use based on classification performance and learning speed. No classifier consistently worked best for all users. So, one might wish to consider other criteria to select the most suitable classifier. In the final application classification should be performed efficiently, without taking too much processing capacity. This makes KStar less suited, as classifying new instances can take long, because the dataset grows. Furthermore, the classifier needs to be regularly retrained in order to keep optimally adapted to the current behaviour of the user. From this perspective, the Perceptron approach seems less suited, because training on new data takes very long. Consequently a Decision Tree or Naive Bayes approach seem most suitable for task recognition in practice.

Furthermore, our research revealed that recognizing tasks on basis of computer activity is personal. Users differ in terms of the tasks they perform and how predictable or difficult their task mix is. Moreover, different users seem to have their own individual way of working. Besides the factors analysed here, other factors might be of influence too. Users might for example have different interpretations of what makes up a specific task and in how precisely they label their activities. Within one user, however, there is a general structure which makes task recognition possible. In general, we can state that a classifier can best be trained for one particular user. When the tool is applied to a new user, we face the so-called cold start problem. This problem can be solved by asking the user what he or she is doing at several moments during one week, thus collecting a representative set of annotated data for this user. As little as 2.5 hours (30 instances) of representative training examples is enough to train a good model. After this week the tool could start to recognize this user's tasks.

During our research we also gained some practical insights. First, some users reported that it was difficult to remember what exactly they had been doing. Some participants noted that the mere fact that they labelled their data made them more aware of the tasks they were performing and some mentioned that this made them work more eagerly. So the data collection procedure, although designed to be unobtrusive, might have had some influence on the way of working. Another observation regarding data annotation was that the users were curious and interested in whether the tool would come up with correct labels, especially in the beginning, which motivated them to regularly check the labels. This curiosity and interest could be further exploited, making the annotation and the tool in general fun to use and game like.

Furthermore, we observed in our experiment that users often think in terms of broader goals, not in terms of the specific methods used. This is in line with the differentiation that Heinze (2003) made (in Tahboub, 2006), describing an intentional level and an activity level. One might regard the more detailed description that the task recognizer comes up with as describing the specific activities performed, including all subtasks. Users agreed that they have actually performed these subtasks, but they themselves describe the tasks they performed during a day at a less detailed level, labelling only their intended main tasks. To capture this hierarchy of task labels one could take a series of subtasks over time to label the sequence with the intended main task label. Temporal models like Markov models or conditional random fields could be considered for modelling these sequences, like is done in related research (e.g. Natarajan et al., 2008). In this way, knowledge of tasks in general could be used to improve the classification, e.g. the fact that information seeking is often a subtask for another main task. One might also wish to use more flexible or overlapping time frames in order to find the exact beginning of new tasks. Nevertheless we see no need to make an overly complex model when with a simple model acceptable accuracy can be reached.

Enabling automatic task recognition is a first step of the SWELL project[5]. With a broader view on the context and mental state of the knowledge worker, we aim to

[5] http://www.commit-nl.nl > Smart reasoning systems for well-being at work and at home

provide optimal support to improve well-being at work. Clearly not all work of a knowledge worker can be captured on basis of computer activity, e.g. time spent in meetings, phone calls, talks with colleagues or reading printed documents. To get a more complete view, we intend to make use of other sources of information. For situations when the user is not active on the computer, we can use information from the user's calendar to fill in gaps. We can also use a camera and microphone to get more information about the user's current situation, like talking to colleagues. Moreover the mobile phone can be a very valuable source of information with call logs, and built in accelerometers and GPS to infer movement and location of the user. We also intend to infer the content the user is working on from documents on the computer. Besides that, estimating the mental state of the user is of interest, like the workload and stress level (Koldijk, Neerincx and Kraaij, 2012). With this information, optimal support and coaching could be provided.

CONCLUSIONS

In this paper we have presented task recognition based on computer activities in a real-life setting. Our research has shown that task recognition on the basis of PC activity is challenging but feasible.

First, task recognition involves more than a simple one-to-one mapping between an application and a task. This is due to interleaved activities, switches to subtasks and a mix of applications used that determine the task performed by the user.

Second, task recognition is very personal. Different users have different work styles and task mixes. Nevertheless, we saw that on an individual basis, the classifiers we used learn to recognize tasks quite fast, yielding a performance up to 80% which is reasonable high, considering the 12 possible task labels that are used.

Third, unlike other research, in which clearly structured tasks were modelled (see e.g. Natarajan et al., 2008), our research has shown that task recognition also works for less structured tasks and more spontaneous activity, since our results were obtained using realistic data.

Fourth, comparison of several classifiers revealed that there is not one classifier that clearly works best in this domain.

Finally, since different users show different patterns of behaviour when performing a task, the classification model should be trained for each specific user to yield optimal task recognition. We concluded that no more than 2.5 hours (30 instances) of representative training examples is required to train a good model.

ACKNOWLEDGMENTS

We thank Iris van Rooij for her support and ideas.

This publication was supported by the Dutch national program COMMIT (project P7 SWELL).

REFERENCES

Albrecht, D., Zukerman, I., Nicholson, A., & Bud, A. (1997). Towards a Bayesian model for keyhole plan recognition in large domains. In *Proceedings of the Sixth International Conference on User Modeling*.

Czerwinski, M., Horvitz, E., & Wilhite, S. (2004). A diary study of task switching and interruptions. In *Chi '04: Proceedings of the sigchi conference on human factors in computing systems.*

Goldman, R. P., Geib, C. W., & Miller, C. A. (1999). A new model of plan recognition. In *Proceedings of the fifteenth conference on uncertainty in artificial intelligence*.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: an update. *SIGKDD Explorations Newsletter, 11* , 10-18.

Johnson, S. M., & White, G. (1971). Self-observation as an agent of behavioral change. *Behavior Therapy, 2* (4), 488 - 497.

Koldijk, S. (2011). *Look what you've done! Task recognition based on PC activities* (Masters' thesis). Radboud University, Nijmegen, The Netherlands.

Koldijk, S., van Staalduinen, M., Raaijmakers, S., van Rooij, I., & Kraaij, W. (2011). Activity-logging for self-coaching of knowledge workers. In *2nd workshop on information access for personal media archives.*

Koldijk, S., Neerincx, M., Kraaij, W. (2012). Unobtrusively measuring stress and workload of knowledge workers. *Proceedings of Measuring Behavior.*

Natarajan, S., Bui, H. H., Tadepalli, P., Kersting, K., & Wong, W.-K. (2008). Logical hierarchical hidden Markov models for modeling user activities. In *Proceedings of the 18th international conference on inductive logic programming.*

Rath, A. S., Devaurs, D., & Lindstaedt, S. N. (2009). Uico: an ontology-based user interaction context model for automatic task detection on the computer desktop. In *Ciao '09: Proceedings of the 1st workshop on context, information and ontologies.*

Richman, G. S., Riordan, M. R., Reiss, M. L., Pyles, D. A., & Bailey, J. S. (1988). The effects of self-monitoring and supervisor feedback on staff performance in a residential setting. *Journal of Applied Behavioral Analysis, 21* (4), 401-409.

Tahboub, K. (2006). Intelligent human-machine interaction based on dynamic Bayesian networks probabilistic intention recognition. *Journal of Intelligent & Robotic Systems, 45* , 31-52.