Look what you've done!
Task recognition based on PC activities

Saskia Koldijk
Radboud University Nijmegen
The Netherlands
SaskiaKoldijk@student.ru.nl
s0610038

June 2011

Supervisors:

Iris van Rooij
Donders Institute for Brain, Cognition, and Behaviour
Radboud University Nijmegen
i.vanRooij@donders.ru.nl

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

Wessel Kraaij
Institute for Computing and Information Sciences
Radboud University Nijmegen
w.kraaij@cs.ru.nl

Contents

Preface

This master thesis is the result of my internship at TNO. During 9 months I worked in the department of multimedia technology in Delft which works on IT solutions and services.

My internship was a precursor to the project 'User Centric Reasoning for Well-working' (UCR4W), which will investigate the key determinants for well-being at work. One of the guiding hypotheses of the project is that logging the activities of knowledge workers can be the basis for an effective computer based coach. In this thesis work regarding the automatic detection of tasks is presented. Part of this work was presented on the IAPMA workshop in Dublin in April 2011. Work performed in the context of the TNO intern knowledge project 'well being at work' can be found in the appendix.

Doing my research project at a TNO gave me a deeper insight in working in a company. Working in a user centered way and using scientific research as basis for developing an application was a new and interesting approach to me. I want to thank Wessel Kraaij for giving me the chance to do my internship at TNO and providing me this interesting project to work on, in which I was able to apply a lot of my knowledge gained during my study of artificial intelligence.

A big thank to Mark van Staalduinen for the weekly meetings in which he provided guidance and many useful tips. With his sharp questions and positive attitude he motivated me to get to the bottom of things. I also want to thank my other colleagues for their collaboration, using my tool to collect data and helping me as experts of various fields. Last but not least I want to thank my third supervisor Iris van Rooij for her trust in me, spontaneously having time and guiding me in scientific working.

Besides working in a company also living in Delft was new to me. My new flat mates made me immediately feel at home. Thanks for the nice evenings and doing the kooking when I was too exhausted! Also thanks to my parents and sister for all the 'gezelligheid' a weekend back home. Finally I want to thank all my friends for their interest and fun time together. Special thanks to Jered for helping me order all my thoughts and Oli for always being there for me.

Abstract

Nowadays many people do their work on PCs. Due to interruptions and task switches it is quite common that at the end of a working day these people experience a feeling of having lost track of what they have been doing during the day. In this thesis a tool was developed which automatically recognizes the tasks a user is performing and presents these in an overview. It was investigated which task labels humans intuitively use and in how far it is possible to recognize these tasks on the basis of low level computer activity, like used applications and clicking and typing behavior. Results show that after only a few hours of training data a reasonable classification accuracy can be reached. Comparison of several classification approaches reveals that there is not one classifier that suits all users best in terms of classification performance and learning speed. Individual differences, due to mix of performed tasks and the individual work style, indicate that the tool should be personalized.

Chapter 1

Introduction

Nowadays many people are knowledge workers, spending most of their working time at the PC. They are coordinating different activities in several project contexts. The information overload knowledge workers experience and the amount of context switches they make can be a threat for their productivity and well-being at work.

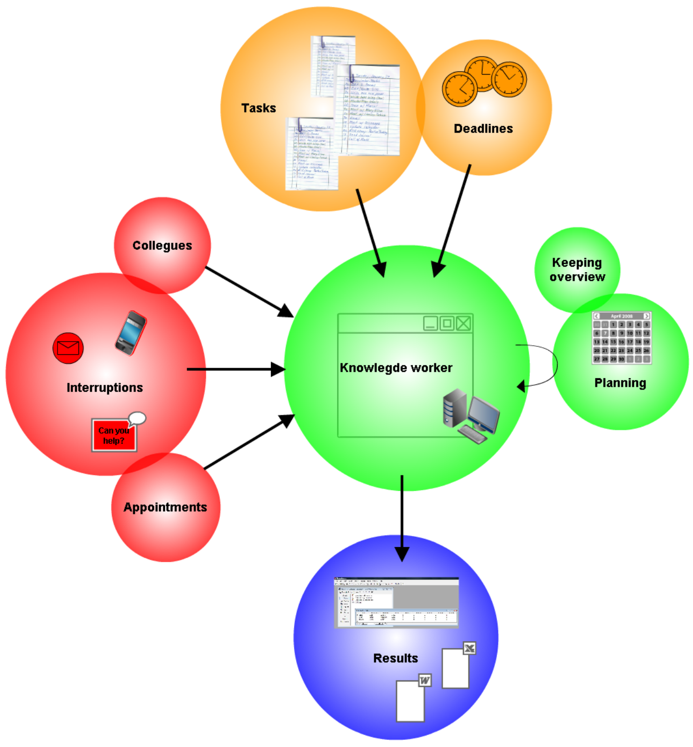


Figure 1.1: Schematic view of demands and influences on the knowledge worker

Let us consider the following scenario: Imagine a typical working day as a knowledge worker. You have different projects running and today your plan is to work on project A and B. For project A you do some Internet search and start typing a document. While you are busy a colleague asks your help for project C. You interrupt your work to help her search for some information. When a mail about project B arrives you decide to switch to this project as it is quite urgent to finish the required document. Suddenly you notice it is 5 o' clock and you did not finish your work on project A as planned. You might start wondering: How did I spend my time today?

As figure 1.1 illustrates knowledge workers typically have various tasks and deadlines and they have to produce results. The possibility to easily switch between tasks makes working very fragmented as the example shows. The 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). Knowledge workers typically have to self-manage their work and make a good planning in order to be able to accomplish all their tasks. This way of working easily causes feelings of stress and it is difficult to keep an overview over what one has done over the course of a day, weeks or even months.

Research aim and motivation

The aim of this research project is to design a system that can automatically recognize and log the tasks a knowledge worker is performing. A study by Czerwinski et al. (2004) has shown that knowledge workers spent 13% of their working time at tracking their own tasks. This indicates the importance of keeping an overview over performed tasks. Automating this process would be of great benefit for the working process. A system that could monitor and provide overviews of performed tasks could be used in a feedback scenario helping the knowledge worker with self-coaching his or her work. By presenting an overview of activities at the end of the day or week and giving an indication of the number of task switches the tool could act as a mirror creating awareness about ones way of working as depicted in figure 1.2.

Figure 1.2: Giving the user feedback about his or her working day could work as mirror creating awareness.

By looking back at the behavior the knowledge worker will get a better overview of the tasks performed and might even get more grip on his or her work style and improve it. Cognitive load and stress could be diminished. More awareness of ones own working process might also have beneficial effects on the on-task behavior and adherence to scheduled activities (Richman, Riordan, Reiss, Pyles, & Bailey, 1988). A study by Johnson and White (1971) showed that mere self-observation had the effect of changing behavior in the desired direction.

Since knowledge workers rely on software for communication, information gathering, document creation and work planning, a vast collection of data can be recorded on the digital devices that are used such as PC, PDA etc. These data include the end-products of a worker's task, but also traces left behind of the work-process itself. The activity of workers can be studied in even greater detail by capturing mouse motion, click events, key presses and active window changes. In our work we want to use these unobtrusive measures to infer on a more abstract task-related level what a user has been doing.

Besides providing overviews, the recognition of tasks is a useful first step enabling other applications. Given a typical pattern of behavior a user shows for some task, deviations from usual behavior could be detected. In this way the system could signal for example stress or a lack of concentration when the number of backspaces is anomalously high for a given task. The system could then attend the user to this or even act as a coach, intervening or motivating him or her. The system could also be extended with automatic annotation by content, which could then be effectively used to assist humans in finding back information.

Research questions

For realizing automatic task recognition certain questions need to be answered.

First of all on which level of abstraction would humans describe their own PC activities? Intuitively a description like 'I spent a lot of time moving my mouse and only a little on using the keyboard today' seems not very appropriate. A more psychological intuitive description could be: 'I started writing a document, until a question arose. I wrote a mail to get a clearer picture, then I continued with the document.'

Moreover, to be able to recognize tasks performed at the PC we need to get a clear picture of the characteristics of this specific domain. How do knowledge workers work at their PC? As we want the system to be non intrusive, we need to investigate which basic PC actions are possible to log and which are actually good indicators for specific tasks.

After this knowledge has been gained our final question is: is it possible for the computer to use simple activity logs to infer tasks on the same description level as the human uses? Different modeling techniques will be considered and tested for their appropriateness.

1.1 Related work

Some systems that provide the user overviews of computer activity already exist (e.g. Slife[1], RescueTime[2]), but they present low-level data in the form of time spent per

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

application and browsed websites. To be interpretable on a higher level they require the user to link applications or websites to self defined categories. In our research we want to put minimal effort on the users and try to automatically detect typical patterns of behavior that are indicative of a particular task. Besides application information we will exploit mouse and keyboard activity to infer what specific task a user is performing.

In the field of action understanding there has already been done much theoretical and applied research. In the more theoretical studies models are made of the way in which we humans recognize the goals of other humans (e.g. Baker, Saxe, and Tenenbaum (2009) or Blokpoel, Kwisthout, and van der Weide (2010)). These provide a basis or inspiration for models with which computers can infer the goals of humans.

In more applied settings several activities are recognized by tracking events with various sensors, for example activities of daily living (Duong, Bui, Phung, & Venkatesh, 2005), activities in an adventure game (Albrecht, Zukerman, Nicholson, & Bud, 1997) or computer activities, like form filling or planning a meeting (Rath, Devaurs, & Lindstaedt, 2009). Common among this research is that the modeled activities have clear structures, involving predefined steps or even underlying plan hierarchies (see Natarajan, Bui, Tadepalli, Kersting, and Wong (2008) for some nice examples). The models used are therefore often logical models assuming a plan library (Goldman, Geib, & Miller, 1999) or Markov models, modeling the sequence of actions in time (e.g. Albrecht et al., 1997). The recognition of knowledge workers tasks on basis of PC activities is a new domain with different characteristics, with tasks being less structured and task sequences being more spontaneous, as we will explain later on.

Some research on detecting patterns in user activity on PCs has already been done. Some systems aim toward improving the human computer interaction, like initiating help when the user is stuck like in Horvitz, Breese, Heckerman, Hovel, and Rommelse (1998). Other work aims at organizing accessed files and retrieving relevant information just in time, like for example association of files with tasks (Shen, Li, Dietterich, & Herlocker, 2006) or pro-actively forming Google-queries on basis of the topic you are writing about (Rath et al., 2008). In general the described applications simply focus on the detection of a specific kind of activity or information to trigger certain actions within the application. In our study we want to use the available information to interpret user behavior on a higher description level in order to automatically make a humanly understandable overview of tasks.

Our intent is to support knowledge workers by giving them more awareness about their working process and in this way help them improve their performance. To our knowledge thus far no such system has been made. Similar, but still different, are the SWISH system by Oliver, Smith, Thakkar, and Surendran (2006), in which the window relatedness and switching behavior is used to detect content clusters which can be used as support for task management, or the TaskTracer system made by Dragunov et al. (2005), in which the user can associate activities with certain tasks in order to easily access records of past activities and restore task contexts. The PAL project, with the subsystems CALO (Myers et al., 2007) and IRIS (Cheyer, Park, & Giuli, 2006) is meant to ease the working process of a knowledge worker by presenting contextual suggestions or helping with time and task management. However automatic task recognition is not (yet) applied in these systems. Myers et al. (2007) themselves suggest that activity recognition would be a welcome addition to identify opportunities were the system could intervene to assist, or to decide when and how to interact with the user.

Structure of the thesis

The remainder of this thesis is structured as follows. In Chapter 2 the requirements for task recognition in the setting of supporting a knowledge worker will be presented. Some theoretical background will be considered in Chapter 3. In Chapter 4 we will outline the framework for inferring tasks form basic PC activity, describing the used task labels, features and classifiers. How we evaluated the task recognition component is explained in Chapter 5, followed by the experiments and results in Chapter 6. We provide a discussion about our insight gained on automatic task recognition in Chapter 7. The thesis finishes with a conclusion in Chapter 8.

Chapter 2

User requirements

In this chapter several requirements will be outlined that have to be considered when making a tool that recognizes knowledge workers tasks. First we will present results of a questionnaire that was set up to investigate the manner of working and specific needs of the knowledge worker. Then we will present some considerations regarding privacy.

2.1 Results of the questionnaire

Taking a user centered approach, a questionnaire was held among TNO employees in order to investigate their working style and preferences for task recognition software. The questionnaire was held online and a link was mailed to all TNO employees in Delft and Groningen. The answers gave us direction on the focus of the project. The five different aspects addressed and the most interesting conclusions are presented now.

The typical knowledge worker

The first set of questions addressed the background of the respondents, including department, function, the amount of time spent at the PC, the number of projects and deadlines and the experienced autonomy.

In total 47 TNO employees with various backgrounds and different functions filled in the complete questionnaire, which provided a broad overview on knowledge workers way of working. The results confirm that the knowledge workers are involved in several different projects at a time (avg: 5.3) and have to manage various deadlines per month (avg: 10.5), which indicates that good self management plays an important role and a tool supporting them would be of benefit. The knowledge workers spend a great amount of their working time (avg: 66.7%) at the PC, which indicates that much information about the working day can be gained from analyzing their PC activities.

The asked knowledge workers stated that they are very autonomous in managing their working time. Nevertheless, results indicate that they are not as skilled as one might expect in making good time estimates. This may point toward a problem that knowledge workers face. The feedback of the software might be useful for them to make better time estimates and in this way improve their management of working time.

The knowledge workers tasks and working style

The next set of questions investigated the tasks the knowledge worker performs and their way of working.

The answers show that the knowledge workers spend most of their time on projects (40%). Another big amount of time is spent on meetings (19%) and mailing (14%), mostly also performed for projects. Notable is that 5% of the time is spent on task tracking (see Figure 2.1). There are clear differences in working day activities between the functional roles of consultants and innovators. Innovators spend more time on project work and consultants more time on email and meetings. This may point toward the existence of different typical user profiles.

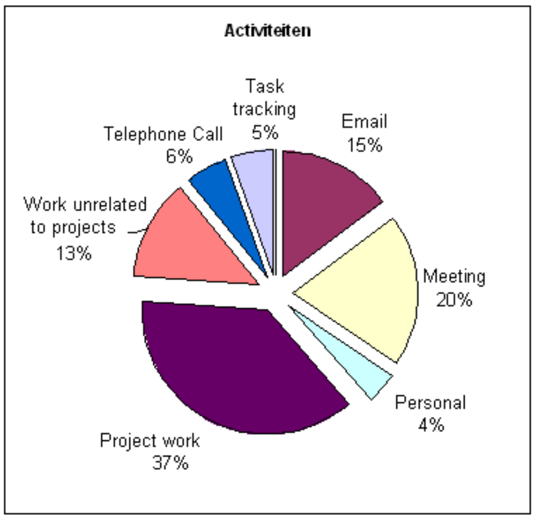


Figure 2.1: Percentage of time knowledge workers indicate to spend on specific activities. Note: Task categories of Czerwinski et al. (2004) used.

There seems to be no clear preference for working style among knowledge workers. Some prefer to focus on one task whereas others like to switch tasks. This is important to know for individualizing the software toward the specific user. The same holds for interruptions. There are individual differences whether they are perceived as annoying or not. The knowledge worker in general faces a tension between working focused on her task versus letting herself distract. This might reflect the tension between individually working on projects versus communicating with colleagues and networking, which is also important for their work.

As can be seen in figure 2.2 a great extent of the knowledge workers task switching behavior is determined by outer influences (email, information requests, appointments, telephone calls together 67%) and only a small part is self determined (33%). The initiation of activities by outer demands may cause stress, fragmentation of work and loss of overview. This indicates an opportunity for software that automatically logs all activities and gives overview.

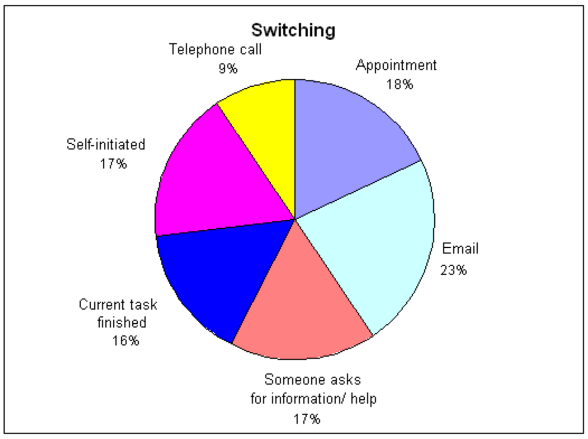


Figure 2.2: Reasons why knowledge workers switch tasks. Note: Switching reasons comparable to Czerwinski et al. (2004) used.

The preferences for task recognition

The next questions considered the preferences the knowledge worker has for the task recognition software, which includes on what level of interpretation the actions should be labeled (from 'simple application logs' to 'linking activities to projects on basis of content') and in which way the activities should finally be presented to the user (from 'detailed time lines' to 'more general overviews').

Recognition on the level of tasks is perceived as more useful than simple application logs. As simple application logging systems already exist, this finding encourages our idea that a system that can interpret behavior on an higher level is a useful new application. Linking actions to projects is found to be useful which means that in future effort should be made toward detecting the project to which a specific task belongs.

Presenting the user detailed action overviews is not rated as useful. A coarse categorization of activities, presenting the projects worked on, and linking activities to projects seem most useful. This indicates that a tool that simply logs activities and presents them to the user is not desirable. It is important that the system aggregates and interprets the logs and optimally links actions to projects to yield the user useful information, which is a great challenge.

The way they plan and manage their work now

Another block of questions addressed the way knowledge workers plan and manage their work now.

The questionnaire shows that many knowledge workers simply work off to-do list (60%) or merely have goals for the day (66%) or week (47%), which indicates a quite flexible working style. Exactly planning when to do what (used by 15%) seems not a very useful or feasible strategy in the work setting of a knowledge worker. 50% of the knowledge workers indicate to work reactively, which is a great amount. This means for many workers the course of actions over a day is not self determined which might cause a lack of overview or a feeling of stress.

Analyzing the results in more detail revealed that there were differences in planning methods used for innovators and consultants (see Figure 2.3). Innovators more often set goals per day or week, whereas consultants more often work reactively. This can be explained by the analysis of the working day activities. It was shown that innovators spend more time on project work which probably requires a good planning and consultants spend more time on email and meetings which can best be handled reactively. The differences in planning method suggest that innovators and consultants might have different requirements for a supporting software.

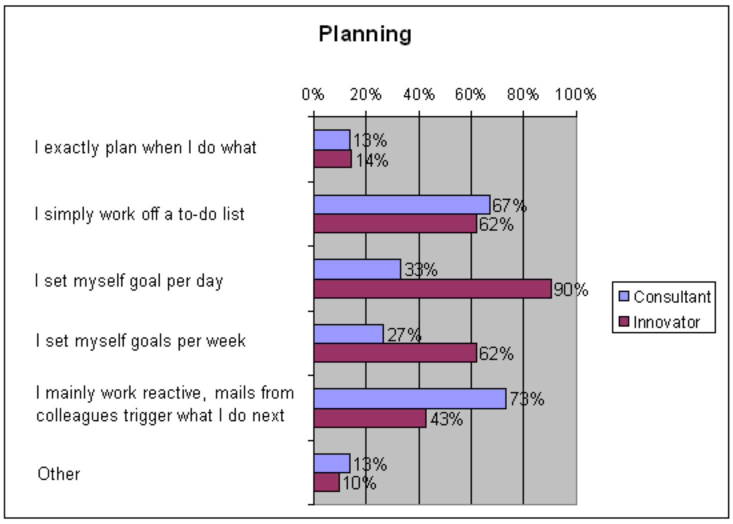


Figure 2.3: How do you plan your work, consultants vs. innovators. Note: More than one answer possible.

Most knowledge workers evaluate their working day in relation to their planning (70%). A system that could help them by providing overviews of activities or visualizations of plans in relation to actual behavior would probably be very valuable and useful for them. Particularly many innovators evaluate their day (81%) which indicates that especially for their type of work it is important to keep track of performed activities.

Most reasons people give for evaluating a working day have to do with keeping overview and planning. This expresses the importance for knowledge workers to keep overview over their activities and being able to coordinate and plan. At the same time it shows demands or aims for the software: It should give the knowledge worker insights in his working style and support him in his self management. Most people that not evaluate their working day see no reason to do so and some do not make time for it. On the one hand this could mean that these people will not use a system that can provide them overviews of their days. On the other hand it could mean that when the system makes looking back at a day or week an easy task these people could see the benefits and actually gain new insights.

How the system could help

Finally we asked the knowledge workers in which way the system could help them, which includes the purpose of the system (from ‘giving the worker himself daily overviews’ to ‘giving others real-time monitoring access’) and the possible benefits of the system (from ‘gaining more insights in ones working style’ to ‘enabling better planning’).

Answers revealed that the main aim of the system should be to give the user himself insights, especially in the form of weekly overviews. No effort has to be made toward providing real-time access of data by others as this is not valued. Moreover one has to care about privacy issues as there is a preference for better not sharing the information with colleagues or the manager.

Respondents see most benefits of the system in improving their own working and managing skills. They do not see benefits for coordinating the work of different people. So there seems to be no need to make interfaces to share information with others.

From the responses we conclude that for a tool to be usable for support, the captured activity data should be aggregated to a higher level in order to provide the user with valuable information. The recognition of the task a user is performing is a useful first step toward providing the user understandable feedback and insights about his working process.

2.2 Privacy

Finally an important factor is privacy. Users might be working on confidential documents or might prefer to not share information about their work with others. Logging and task information should therefore be stored locally on the users’ PC, unavailable for others. For task recognition extracted features should to some extent be abstracted and made anonymous and use of information about text typed should be avoided.

Chapter 3

Specifics of our domain

In this chapter we will analyze the specifics of our domain: the knowledge worker working at his PC in an environment with other colleagues. First we will present some theory about the study of context which is important for gaining a good view on the knowledge workers' task performance. The consequences that the specifics of this domain have for task recognition will be outlined. Then we will present some remarks considering the interaction with the system.

3.1 Studying the knowledge worker in her context

Various research (see Nardi, 1996) has shown that it is not possible to fully understand how people work without considering other people and artifacts they use for accomplishing their task. In our case, it is important to not only consider the knowledge worker and the tasks he or she aims to perform, but also the PC and outer influences, like incoming mails or calls that influence the work. There are different approaches to the study of context (Nardi, 1996).

The 'distributed cognition' approach states that the unit of analysis should be the individuals and the artifacts they use, which together form a cognitive system. For our domain this means that the knowledge workers in interaction with the used devices should be analyzed. The different people and devices work not independently of each other, but can be seen as one system in which all parts influence each other.

In the 'situated action models' approach it is emphasized that the environment is an important shaper of human action. Humans often act in a responsive, flexible and improvisatory way. The specific structuring of an activity often grows out of the situation instead of being planned on forehand. It is stated that goals are merely 'retrospective reconstructions' and that 'plans are an artifact of our reasoning about action, not the generative mechanism of action' (see Nardi, 1996). The fact that humans can easily explain their actions does not mean that all actions are indeed well planned at forehand. In our domain this means that the flow of activity of the knowledge worker might not follow clear plans and structures, but arises as he or she is interacting with his or her environment, yielding a rather spontaneous or unstructured sequence of actions.

In the 'activity theory' approach it is stated that the objective the human has stays fixed, but the goals, actions and operation performed change as conditions change. In our case this means that the knowledge worker might have one specific goal in mind, but how he or she actually accomplishes this goal might change over time depending on the encountered context.

All three approaches have in common that the human is not an encapsulated processing device, that strictly works out plans. Interactions with the computer, as well as other people and the situation influence the way of working. This causes a great variance and flexibility in human task completion which poses a big challenge for modeling activities of the knowledge worker. To tackle this challenge we will consider the tips Nardi (1996) gives the field of human computer interaction for dealing with the complexity of user context:

- *Attention to broad patterns of activity, rather than narrow episodic fragments that fail to reveal the overall direction and import of an activity*

  As all approaches to context reveal, the specific situation is of great influence on the task performance. Users accomplishing similar tasks may work differently and even for one user there might be variation in task completion due to his or her current context. For our task recognition framework the lack of a common structure in task completion might argue against modeling sequences of actions in time. One should abstract away from specific action series and aim to recognize stable commonalities among a task. A sort of 'bag of activity' approach might be more suitable, in which all activity within a frame in time is summed up and taken as indicator of the task done during that frame.

- *A research time frame long enough to understand users' objectives and changes in objectives*

  The specific time frame used for recognizing activities is an important factor. A too short time frame and too fine grained analysis might fail to identify global patterns of main activities and get stuck in details. A longer time frame can yield a more stable basis for task recognition, although a too long time frame might miss important short termed activities. The time frame of analysis should thus be well chosen.

- *A commitment to understanding things from users' points of view*

  When making a tool that is based on activity logs of PC events, one might easily fall to a technology driven approach, focusing on actions that can easily be logged and presented. Instead one should try to take a user centered approach in which the technology is used to support the human in the best possible way. Therefore the goal should be to understand the human view on tasks and actions and let the software adapt to the needs and goals of the user. For task recognition this means that the knowledge workers should be asked when defining a set of suitable tasks to be recognized.

- *The use of a varied set of data collection techniques (interviews, observations)*

  We may ask users about their way of working in interviews or questionnaires. But as theory points out, these descriptions might be retrospective reconstructions that do not really reflect the underlying model of task performance. Therefore it is also important to observe users' PC activities on basis of computer logging data. In this way real life task performance can be analyzed, yielding a more realistic view. For task recognition this means that preferably in an pilot study in which data in a real work setting is collected, the PC activities chosen to log, and their expected potential to discriminate tasks, should be verified.

3.2 Interaction with the system

Besides a robust and user centered task recognition the manner in which the system interacts with the user is also an important factor. As the tool is used during work it should not demand too much attention of the user. The interaction should be as pleasant as possible. Myers et al. (2007) state that the system should be directable, personalizable, teachable and transparent. So in order to work well the system should optimally cooperate with – and adapt to – the user. The tool should provide a means for the user to give feedback without irritating him and it should keep learning. The task recognition module should be quickly trained and learn the behavior of this specific user without much user interaction.

Chapter 4

Task recognition framework

In this chapter we will present the global framework for task recognition. In order to make automatic detection of tasks possible logging of certain low level PC activity is necessary. This activity can be expressed as features as for example the amount of clicks, typed characters or the amount of time a specific application was in focus. As can be seen in figure 4.1 a classifier will be given these features in order to assign a task label to the activity. Much typing and some clicking while running Word could for example yield the task 'writing report' or some typing and much clicking while running the Internet Explorer could be categorized as the task 'searching information'.

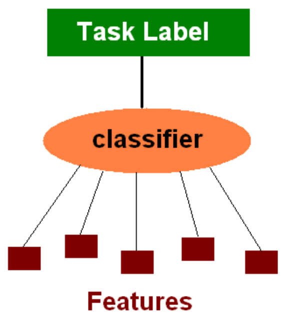


Figure 4.1: The proposed framework for task recognition.

In our approach we chose to handle time in a static manner. The stream of activity is split into time frames of 5 minutes. We expect that this is fine grained enough to capture varying activities, but large enough to average out temporary specifics and gain stable measures of activity within the frame. For each of these time frames the classifier assigns a task label.

The reason for taking such an approach is that we do not expect to gain much from taking temporal sequences into account, as there is no logical order of tasks performed at the computer. As already lined out users are often switching tasks out of outer influences which makes their task sequences rather unpredictable. Therefore we conclude that the current activity is most informative and enough to recognize the task a user is performing. Moreover taking a static approach enables us to use very simple basic modeling approaches, which yield fast task recognition. Last but not least taking a temporal approach would have required a very good ground truth labeling in which each consecutive time frame is labeled by a user. It is unrealistic to expect to get such a good ground truth labeling.

In the following subsections the three components required for task recognition will be defined:

1. A set of suitable task labels
2. A number of useful features gained by logging low level computer activity
3. Different classifiers that map low level activity features to the defined task labels

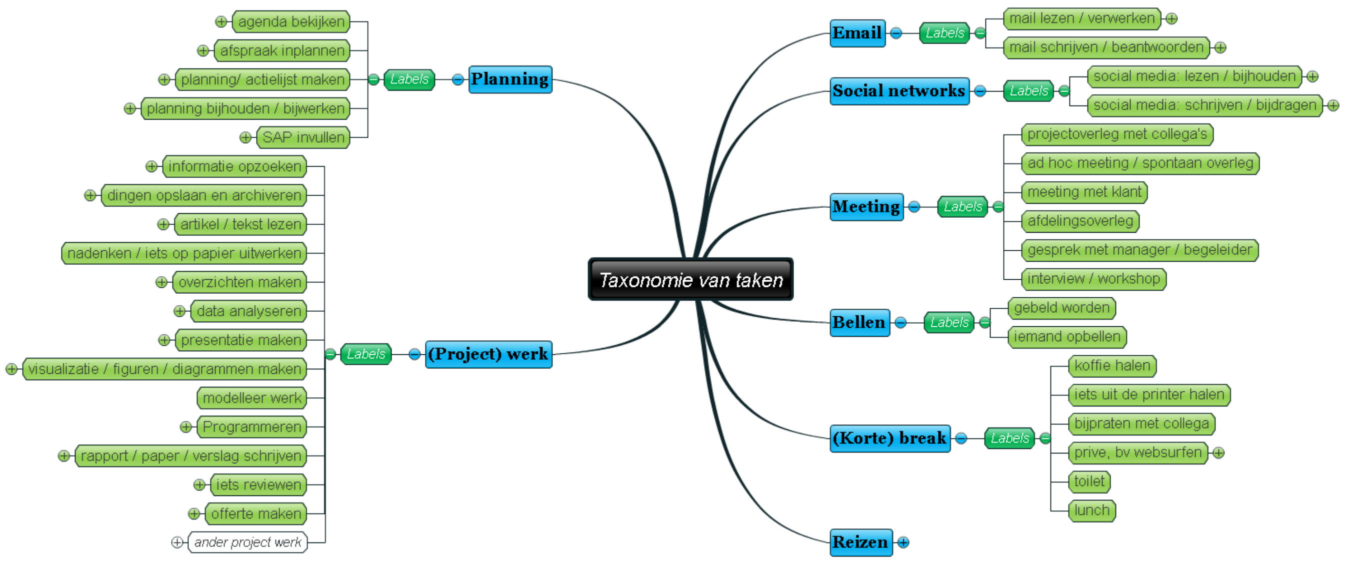


Figure 4.2: Final taxonomy (created on basis of the questionnaire)

4.1 Task labels

The first step is investigating which tasks knowledge workers typically perform in order to define a set of task labels. Several task taxonomies have already been proposed in the literature. The taxonomies about Internet use by Morrison, Pirolli, and Card (2001) indicate that a distinction between actions on three different levels might be appropriate: The method the user adopts, the purpose of his actions and the specific content. In our case the low level PC activity could be seen as method the user adopts in working. On a higher, purpose level one could define several tasks that the user tries to accomplish. The highest level constitutes the content, which in our case could be the specific project context a task is performed for. In this research we will focus on the method the knowledge worker adopts to infer his tasks on the purpose level.

In further research more content related features could be added in order to classify actions further on the content level.

A set of tasks that knowledge workers typically perform was identified on the basis of the questionnaire. The answers to the questions 'What tasks do you perform and how do you use your computer for this?' and 'Describe a typical working day' were manually grouped into sets of similar answers. Task categories clearly arising from the data were email, meeting and planning, which were mentioned by nearly anyone. Depending on the specific work field of the knowledge worker several project work tasks were mentioned, like searching for information, analyzing data, making a presentation or writing a report. Many people also listed phoning, traveling, use of social media or breaks to get coffee, chat with colleagues, do some private Internet browsing, or have lunch. The complete taxonomy with labels that knowledge workers intuitively used to describe their activities can be seen in Figure 4.2.

Table 4.1: Example responses

'What tasks do you perform and how do you use your computer for this?'
mails (outlook) actielijsten bijwerken (outlook) lezen (kleinere) stukken (word, excel, ppt) maken presentaties (ppt) notities (word, ppt) financieel zaken (SAP, webfocus, excel)
Presentaties: powerpoint, webbrowser, visio Rapporten/offertes: Word, powerpoint, visio, paint, browser Research/kennis: Webbrowser, filemanager, powerpoint, word
email lezen thesis schrijven info zoeken planning bijhouden
Ik heb altijd zeer veel vensters open staan. Vaak is het navigeren tussen word/powerpoint en firefox en andere word/powerpoint/pdf documenten.

What became clear to us is that humans 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. In an initial taxonomy we made on basis of our intuitions the root was split up into the application the user accesses on the PC: the Internet, a document, the file system or mail. For each application we then listed the specific possible purposes. These initial ideas turned out to be too technically focused. Table 4.1 gives an impression of typical questionnaire answers. As you can see, the tasks are in focus, the applications used depend on these tasks. Important to note is that some applications, like PowerPoint, are used in several tasks. Therefore task recognition is no simple one-to-one mapping between an application and a task. Users also seem to switch between different applications while executing one task, which is made explicit by the fourth respondent. This should also be taken in mind when designing the recognition software.

So all in all the questionnaire helped us making a human centered approach to task labeling. The appropriateness of the identified task labels was confirmed by several knowledge workers. From all the identified task types, the tasks performed at the computer as listed in table 4.2 were finally selected for automatic task recognition.

Table 4.2: Overview of task labels

Task label
read mail
write mail
organize/archive data
plan
make presentation
create visualization
program
write report/paper
search information
read article/text
make overview
analyze data

4.2 Features

In order enable automatic task recognition user data must be automatically logged. From this raw user data useful features should be extracted with which the recognition module can discriminate between tasks. In this section the logging tool and the extracted features are presented.

Logging

The most obvious activities to log for inferring users' tasks are mouse and keyboard actions as well as the application that users use. Several different applications to record these activities can be found on the web. After some experimentation the software uLog by Noldus[1] seemed most appropriate for our purposes. It logs many types of low level activities and stores everything in an easily accessible xml format.

The logging files contain the following (usable) information:

- The event noticed, incl. specific details:
  - Mouse
    - EventAction: clicked, double clicked, wheel turned
    - MouseButton: left, right
    - MouseCursorX: 91, ...
    - MouseCursorY: 125, ...
    - MouseWheelTurnAmount: 0, ...
  - Keyboard
    - EventAction: character, special key
    - KeyboardValue: 'b', 'a'[2], space, ...
  - Other
    - EventAction: Window activated, MessageBox activated, Application started, Application exited

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

[2] all characters are finally stored as 'X' in the database because of privacy reasons

- Where the event was (the control belonging to this event)
  - ControlType: Button, Menuitem, List, Pane, Window, Document, Header, Edit, Combobox (url)
  - ControlCaption: 'New', ...
  - ControlApplication: wordpad, ...
  - ControlWindowText: 'Labelling.txt - WordPad'
- Timestamp of the event

Table 4.3: Overview of extracted features

Feature name	Description
user	the user who logged and labeled the data
daytime	time of the day (as hour, i.e. 9-18)
clicks	# clicks within the time frame (expressed on a 'per minute' basis)
scrolls	# scrolls p.m.
characters	# characters typed p.m.
specialKeys	# special keys typed p.m.
backspaces	# backspaces (inc. 'delete'-key) p.m.
spaces	# spaces typed p.m.
switches	# switches between applications
nrApps	# different applications used within the time frame
mainApp	application that was most of the time in focus
time	% time this mainApp was in focus
internet	% time that an Internet application had focus (iexplorer, firefox...) within the time frame
programApp	% time that a programming application had focus (eclipse, cmd...)
editor	% time that an editing application had focus (notepad++, wordpad...)
WINWORD	% time that WINWORD had focus within the time frame
OUTLOOK	% time that OUTLOOK had focus
POWERPNT	% time that POWERPNT had focus
AcroRd32	% time that AcroRd32 had focus
explorer	% time that the explorer had focus
mspaint	% time that mspaint had focus
EXCEL	% time that EXCEL had focus
MATLAB	% time that MATLAB had focus

Feature extraction

The raw data captured by the logging tool is processed to extract several features. All these features were determined on basis of 5 minute time frames, which we assume is long enough to average out fluctuations, but fine grained enough to not loose useful information. Mouse features include the number of clicks and scrolls within the time frame. Keyboard features include the amount of characters and special keys typed as well as the number of spaces and backspaces. Application features include the application that was mainly in focus during the five minute time frame and separate application 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 and the specific user (see Table 4.3 for a detailed list of features).

In a pilot study presented in Koldijk, van Staalduinen, Raaijmakers, van Rooij, and Kraaij (2011) data of three knowledge workers was collected during two weeks of their normal work routine. Analysis of the resulting labeled dataset proved the usefulness of these features to discriminate the defined tasks.

4.3 Classifiers

For mapping simple logging features to higher level tasks, several classification approaches are considered. To investigate which of these approaches is most suitable for our domain of task recognition, we intend to compare the performance and learning curves of the different classifier types.

In general a classifier is first provided with a training set of labeled data to train a model. Then the trained classifier can be applied to classify new data and in this way enable automatic task recognition. The chosen classifiers will be described now. For all classifiers we used the implementation in Weka (Hall et al., 2009) with default settings.

KStar

KStar is an instanced based classifier. That means all examples from a dataset for training are simply stored, no specific model is trained. A new data instance is classified by comparing it to the stored examples in order to find the most similar ones. The KStar algorithm implemented in Weka uses an entropy measure for determining the similarity (Cleary & Trigg, 1995). The label of the majority of the neighbors is chosen as task label. This approach is also called nearest neighbor classification.

The advantage of this approach is that arbitrary complex structures in the data can be captured. As disadvantage one could see that no abstraction takes place as no model is trained. All training instances are simply stored, which means that training and retraining this model is fast. Classification however takes time as the dataset grows, because comparison with many instances is required.

Decision Tree

The decision tree J48 is a rule based approach Quinlan (1993). On basis of the examples in the train set, a set of rules is determined that is suitable for splitting up the data into the defined classes. The rules and the order in which they should be applied form a decision tree. A new data instance is classified by checking values of various features with the rules in the order they appear in the tree. The final branch yields the task label.

The advantage of this approach is that it is very simple and the created decision tree can give insight in the structure of the data. A disadvantage could be that the approach is too simple and the model fails in separating all task labels correctly. In general as well training this classifier as classifying new instances is quite fast.

Naive Bayes

Naive Bayes is an probabilistic approach to classification (John & Langley, 1995). In a simple Bayesian network with task label as root and the features as leave nodes it is represented how probable specific values of the features are given that someone performs a particular task. We used the supervised discretization option to split the continuous features into bins of values. Based on the training data all probability distributions were learned. A new data instance is classified by calculating what the most probable task is given the feature values.

The advantage of this approach is that it is quite simple and the Bayesian network yields insights in the structure of the data. An disadvantage could be that the domain could not be modeled in such a structured, probabilistic way. In general as well training this classifier as classifying new instances is quite fast.

Multilayered Perceptron

The Multilayered Perceptron is a neural network approach (Bishop, 2006). This means a network is used with an input layer consisting of a node for each feature, some hidden nodes and an output layer representing the task label. By considering all examples from the train set the strength of the connections between the nodes is learned, causing that connections from some features to tasks become stronger and others weaker. A new data instance is classified by feeding the values of the features into the input layer. Activity then spreads and the most active output node represents the selected task label.

The advantage of this approach is that arbitrary complex structures can be learned, given enough training data. The disadvantage is that training such a classifier requires very much time.

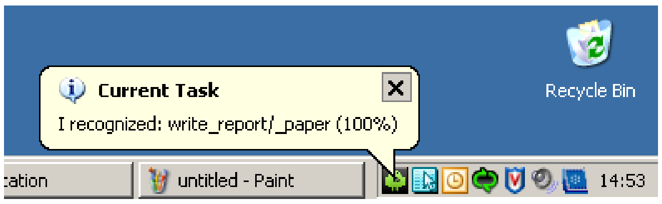
Chapter 5

Evaluation of the Classifiers

Given the defined set of task labels and chosen features it was evaluated how good the different classifiers are in classifying user behavior as a specific task. For this analysis a large annotated data set is required. Therefore a tool to collect annotated data in a user friendly way was made. In this chapter we will first describe the tool in more detail and then outline the approach taken to investigate the performance of the classifiers.

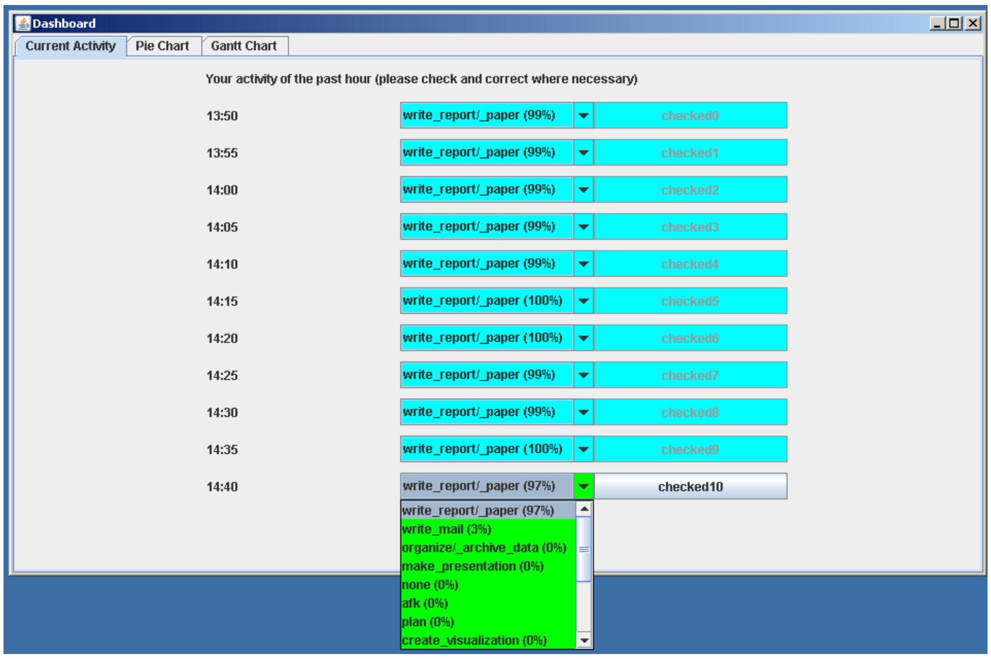
5.1 Tool for collecting annotated data

In order to train classifiers and evaluate their performance a ground truth labeling of the users' activities is needed. A simple pop up reminding users to indicate which task they are currently performing was perceived as very annoying. Therefore we created a more user friendly data annotation tool, which makes the labeling easier by proposing task labels to the user. A classifier trained on the initially collected dataset of our pilot study is used to automatically classify the current five minutes of user activity. The recognized task label is then presented to the user (Figure 5.1). The user can look back at the proposed task labels of the previous hour and confirm or correct them (see Figure5.2). In this way the activity labels can easily be checked whenever the user wishes to. After one hour of new data the classifier is retrained to optimally predict suitable task labels for this user.


Current Task

I recognized: write_report/_paper (100%)

Figure 5.1: Pop-up presenting the recognized task label


Dashboard

Current Activity | Pie Chart | Gantt Chart

Your activity of the past hour (please check and correct where necessary)

13:50	write_report/_paper (99%)	checked0
13:55	write_report/_paper (99%)	checked1
14:00	write_report/_paper (99%)	checked2
14:05	write_report/_paper (99%)	checked3
14:10	write_report/_paper (99%)	checked4
14:15	write_report/_paper (100%)	checked5
14:20	write_report/_paper (100%)	checked6
14:25	write_report/_paper (99%)	checked7
14:30	write_report/_paper (99%)	checked8
14:35	write_report/_paper (100%)	checked9
14:40	write_report/_paper (97%)	checked10

write_report/_paper (97%)
write_mail (3%)
organize/_archive_data (0%)
make_presentation (0%)
none (0%)
afk (0%)
plan (0%)
create_visualization (0%)

Figure 5.2: View to check or correct the automatic labeling

Besides making labeling of activities easier, we added two types of visualizations to make the use of the program more interesting for the users. The first visualization depicts the performed tasks as a pie chart (see Figure 5.3). This gives the knowledge worker the possibility to look back and see on which sort of tasks he or she was mainly working over the days. The second visualization shows the activities of the knowledge worker as a Gantt chart (see Figure 5.4). In this visualization he or she can easily see the course of activities over the day. Our idea was that giving users these visualizations made it more interesting to correctly label their activities and provide them more insights in their way of working.

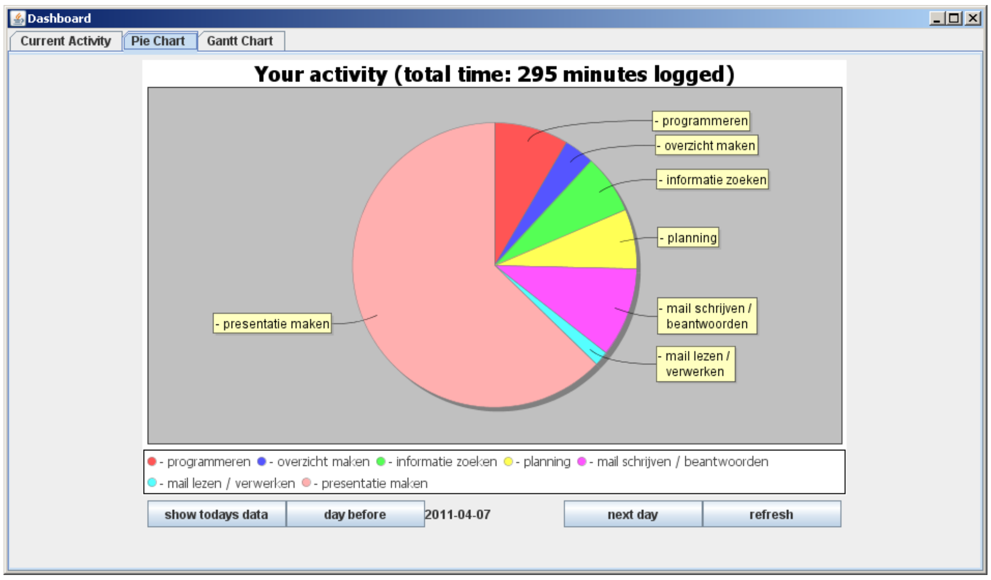

Dashboard
Current Activity
Pie Chart
Gantt Chart
Your activity (total time: 295 minutes logged)
- programmeren
- overzicht maken
- informatie zoeken
- planning
- mail schrijven / beantwoorden
- mail lezen / verwerken
- presentatie maken
● - programmeren ● - overzicht maken ● - informatie zoeken ● - planning ● - mail schrijven / beantwoorden
● - mail lezen / verwerken ● - presentatie maken
show todays data
day before
2011-04-07
next day
refresh

Figure 5.3: Pie chart visualization

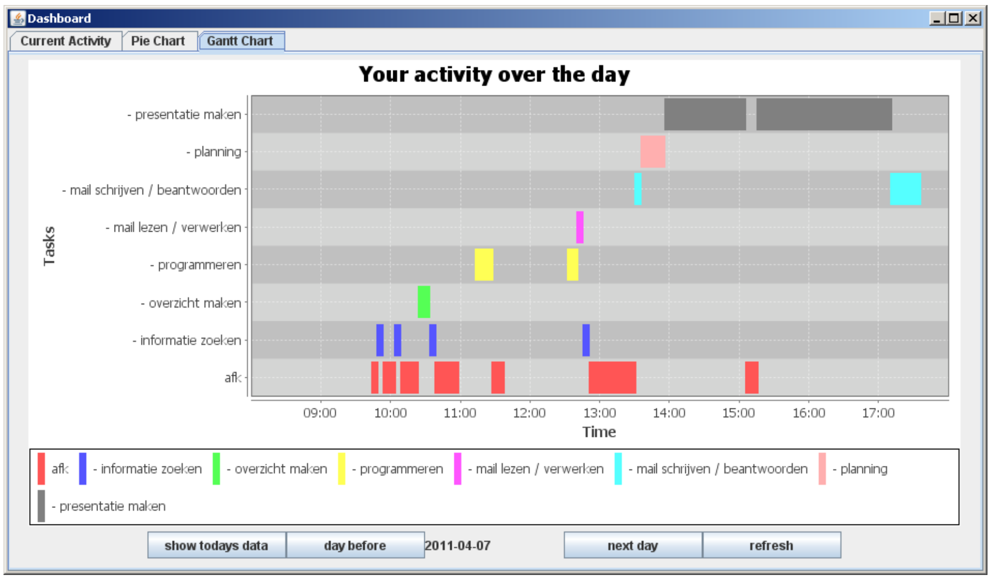


Figure 5.4: Gantt chart visualization

5.2 Approach

Our aim is investigating how precisely the tasks the user is performing can be inferred. Therefore we want to know: What is the classification accuracy of the different classifiers? Moreover we want to find out which classifier is most suitable for our scenario. We are not only interested in the best reachable classification performance, but also how quickly the classifier reaches a high performance level. One of the demands on the final software is that the tool is able to categorize activity quickly, without much training. Therefore a classifier with a steep learning curve is preferable. The time it takes to train or retrain the classifier has also to be considered. Finally we are also interested in whether these results apply in general or if there are differences between users, i.e. if specific classifiers work best for only some users. To summarize, our dependent variables are classification accuracy (percentage correctly classified) and the steepness of the learning curve. Our independent variables are the type of classifier and the specific user.

Our approach is the following: First of all we collect data of different users over several working days. The resulting annotated dataset is then used for our analysis. To simulate the learning process of the classifiers we split each user's final dataset into several parts, containing successively more data. The classifiers are then trained on these parts of the data. In this way we are able to offline simulate how quickly each classifier reaches a good performance.

5.3 Method

In this section the exact method we followed for collecting annotated user data is described.

5.3.1 Participants

Eleven knowledge workers employed at TNO volunteered in our two week long data collection trajectory. All participants typically spent most of their working time at the PC's and carried out a diverse set of typical knowledge worker tasks like mailing or document writing.

5.3.2 Materials

The participants worked at their regular working place on their own Windows desktop computer with mouse and keyboard. The logging tool uLog was installed on the machines in order 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. For task recognition another Java program fetched the current activity data from the database using SPARQL and various classifiers from the machine learning toolkit Weka were used to propose a task label to the user.

5.3.3 Procedure

First of all the required software was installed on the participants computers and shortly explained. The knowledge workers were instructed to start up the software at the beginning of the day and work as usual. While working the data capturing programs ran without attracting attention. Every five minutes the recognition program analyzed the activity data of the user and one of the classifiers was chosen at random to propose 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 proposed task labels and correct them when necessary, either immediately after the pop-up or within one hour via the dashboard view (see Figure 5.2). It was explained that some simple visualizations of activities of the days were automatically made which could be accessed via the dashboard whenever the participant wished to. This made the program more interesting to use, as the participants could get insights in their work process.

Chapter 6

Results

In this chapter we will present all our results. In the first section the results regarding the comparison of different classifiers are presented. Conclusions about which classifier is best for recognizing knowledge workers tasks are drawn. In the second section specific analysis regarding individual differences between users is presented. Conclusions about the way in which users differ from each other and what consequences this has for developing a general task recognition model are outlined. In the last section analysis regarding the chosen features, task labels and framework is presented. Conclusions are drawn to whether improvements can be made regarding the set of features and task labels as well as the static task recognition framework.

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 for our analysis. In table 6.1 the amount of checked labels per user can be seen. As user J and B checked too little labels, their data were excluded from further analysis.

Table 6.1: 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

6.1 Comparison of classifiers

6.1.1 Performance

In order to answer the question which classifier is best in recognizing tasks we used Weka (see Hall et al., 2009) to train and test several classifiers.

The performance of the classifiers was measured as percentage correctly classified instances. For performance evaluation we applied 10 fold cross-validation. This means that a dataset is split into ten parts or folds. Nine of these folds are used for training a classifier model and the remaining fold is used for testing the performance of the classifier. This is repeated ten times so that finally all folds have been test set once and results are averaged over all folds, which yields a stable performance estimate.

To make the estimate even more reliable we ran this whole process ten times and averaged the results over the runs.

Labeling each activity simply as the majority class 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 labeled data of one user at a time was used to train and test a classifier. This was repeated with all nine users' data. The classification performance of each classifier for the various users can be seen in Figure 6.1.

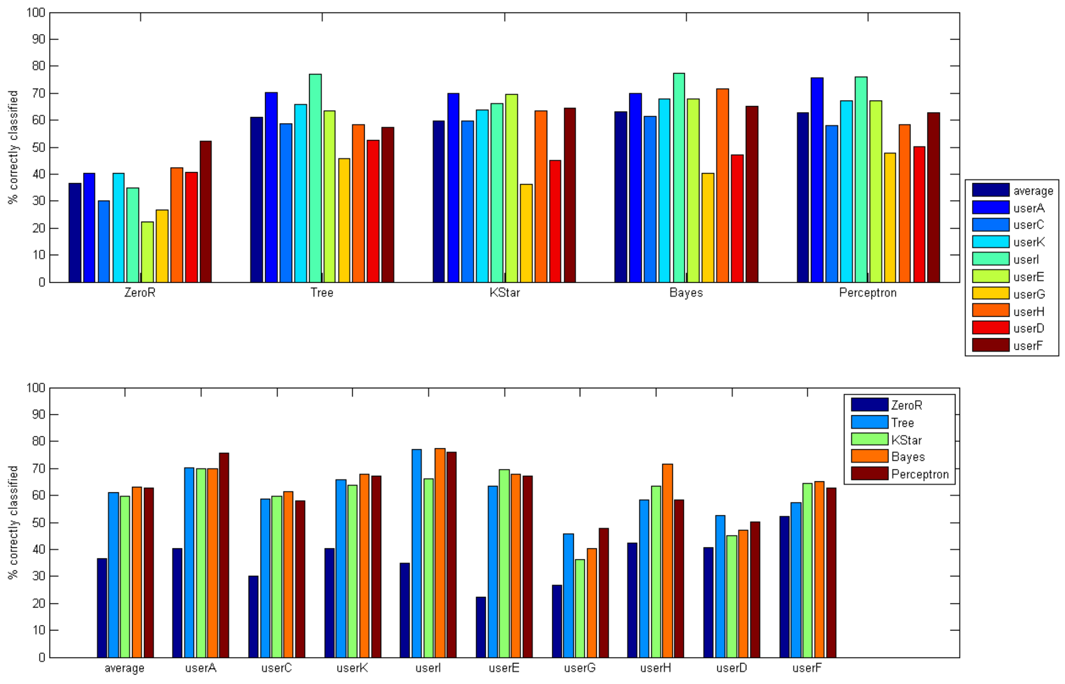


Figure 6.1: Performance of the classifiers. Top: grouped per classifier, bottom: grouped per user. Note: ZeroR provides a baseline. Users ordered on amount of available data – high to low.

As can be seen in the upper plot the performance that can be reached with a classifier varies greatly per user. Performances between as low as 40% and as high as 80% were reached, depending on the specific user. In the lower plot one sees that averaged over all users every classifier is about equally good and reaches a performance of slightly above 60%.

The figure shows that for each user all tested classifiers perform better than baseline (which is given by ZeroR). Statistical analysis proved these differences to be statistically significant for users A, C, K, I, E and H, the other users probably providing too little data. Which classifier reaches the best performance differs per user. For user A the Perceptron is best with 80% classification accuracy, whereas for user H Naive Bayes gives best results with 75% accuracy. For user D Decision Tree classification works best and for user E KStar wins.

From this analysis we can conclude that one cannot say which of the classifiers in general reaches the best performance. As outlined, the different classifiers use very different principles to discriminate 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 will analyze the differences between users in more detail in section 6.2.

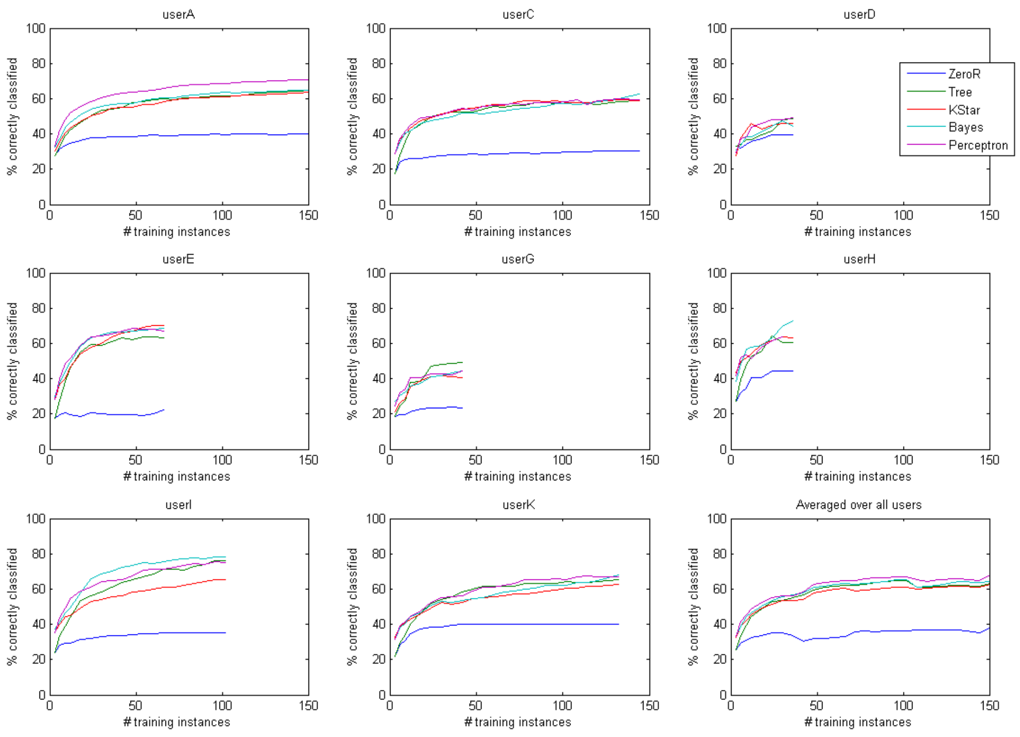


Figure 6.2: Learning curves for the different classifiers, per user. Note: ZeroR provides a baseline.

6.1.2 Learning curves

The next question was which classifier is fastest in learning to classify tasks. We simulated the growths of the data set in order to analyze the learning process of the classifiers. The user's complete data set is first of all split into 10 folds, one of these folds held apart for testing. From the remaining folds data is randomly sampled creating increasingly large training portions. Note that by sampling, each training portion has about the same task label distribution as in the total set. The instances are not added in the order they actually appeared during data collection, as in that case the points in the learning curves would be very dependent on the specific task mix performed at that time.

The first training portion contains 3 sampled data instances, the next 6, the next 9 and the next 12 instances, which equals to one hour of data. From then on the training portion size grows 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).

In Figure 6.2 the learning curves are plotted per user. You see that in general the performance of the classifiers is at 80% of its maximum already after about 30 instances, which is only 2.5 hours of training data. The particular form of learning curves differs per user. For user K all classifiers learn slowly, whereas for user E they all learn quickly. For user I there is much difference between learning curves with Naive Bayes quickly reaching a high performance and KStar performing badly, whereas for user C all curves lie close together, showing no clear winner in learning speed.

From this analysis we can conclude that one cannot say which of the classifiers in general learns quickest. Again specific characteristics of the users seem to have influence on how fast a model is learned and which classifier is most suitable.

6.1.3 Conclusions about which classifier to use

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 issues. In the final application classification should be performed efficiently, without taking too much processing capacity. In light of this the KStar approach seems less suited as classifying new instances can take long as the dataset grows. Furthermore the classifier needs to be regularly retrained in order to keep optimally adapted to the current behavior of the user. In light of this the Perceptron approach seems less suited as training it on new data takes very long. Consequently a Decision Tree or Naive Bayes approach seem most suitable for task recognition in this domain.

6.2 Individual differences

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

- Were 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?

In the next sections we describe our further analysis to answer these questions regarding individual differences.

6.2.1 Differences between users

Given the variance between users in classifier performance we investigated possible causes. A first aspect we analyzed is the variance in the users' tasks. In figure 6.3 the distribution of tasks the knowledge workers performed during the data collection period is pictured. First of all you see that different users do different things, like for example user A is much concerned with programming and report writing, whereas user F mainly plans and makes presentations. This could hint at different functions of the users.

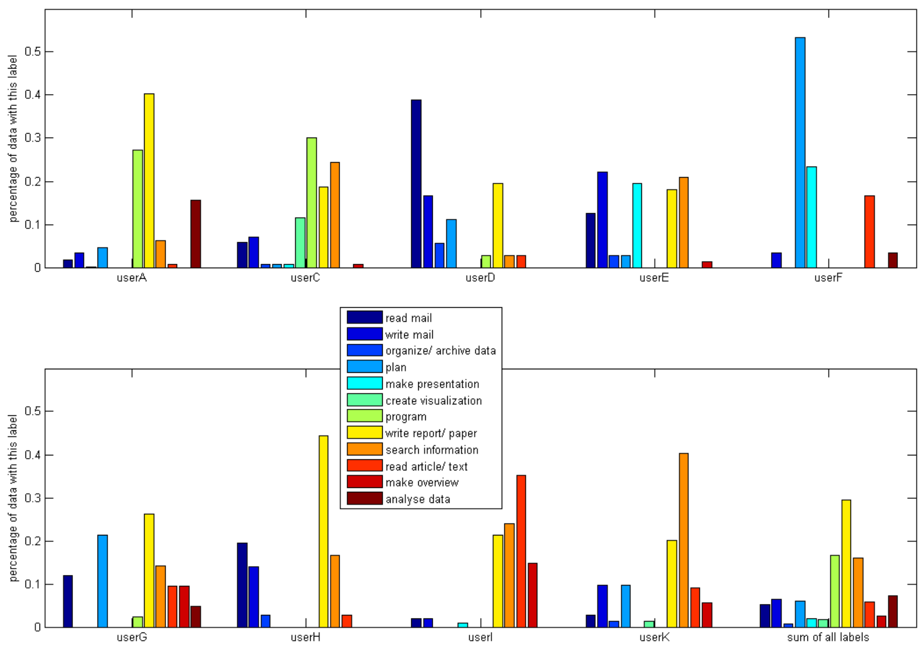


Figure 6.3: The distribution of tasks performed by the different users, as percentage.

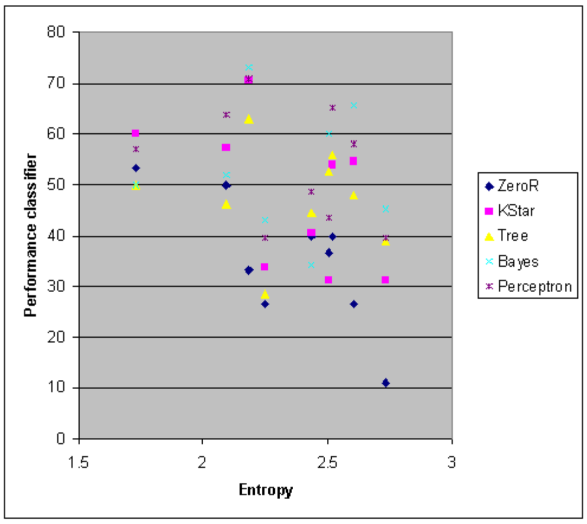


Figure 6.4: The relation between entropy of the task distribution and classification performance.

Moreover, you can see that the distribution is very flat for some users, for example users E and G for whom many tasks have equal likelihood. This makes their tasks difficult to predict. The distributions have a high entropy score. For other users specific tasks have a high likelihood, for example user F mainly plans or user H mainly writes reports. In this case it is easier to predict the task. These distributions have a low entropy score. In figure 6.4 the entropy of the users' task distribution is plotted in relation to the classifiers' performance. There seems to be a slight negative relation between the two. The higher the entropy of the distribution, the lower the classifiers' performance. So a flatter task distribution makes predicting tasks more difficult for the classifiers. This can explain the differences in classification performance between users.

A second aspect we analyzed is the typical pattern of behavior of the users. Boxplots were made depicting the typical amount of clicks and typed characters for specific tasks. In figure 6.5 you see how users differ in behavior for the task of report writing or information seeking. You see that even when users are performing the same task their behavior differs. For example user G in general clicks more often than other users. Moreover he types extraordinary many characters when writing a report and very little when searching for information. 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 as well the task as the specific user have influence on the measured behavior.

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. Users with very structured activity patterns might best be modeled by a rule based Tree approach or a probabilistic Naive Bayes approach. Users displaying a more chaotic pattern of behavior might better be modeled by the nearest neighbor KStar approach.

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

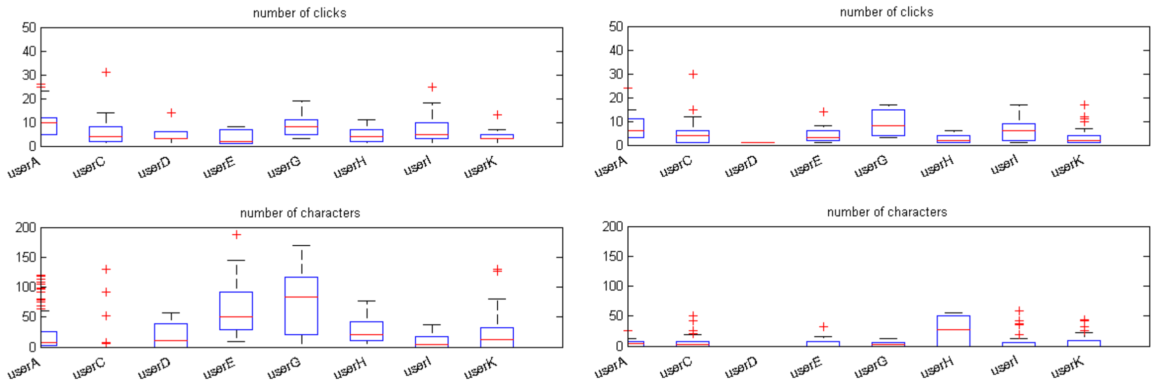


Figure 6.5: Amount of clicking and typed characters compared between users. Left: task 'write report', right: task 'search information'.

6.2.2 Generalizability of the classifiers

Analysis thus far indicates 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 behavior.

To answer this question we first trained a classifier on 90% of user A's data. This classifier was then tested on all other users' data. To get a fair comparison of classifier performance between participants it was decided to use equally large datasets for each user, so 30 instances of each users' data were sampled.

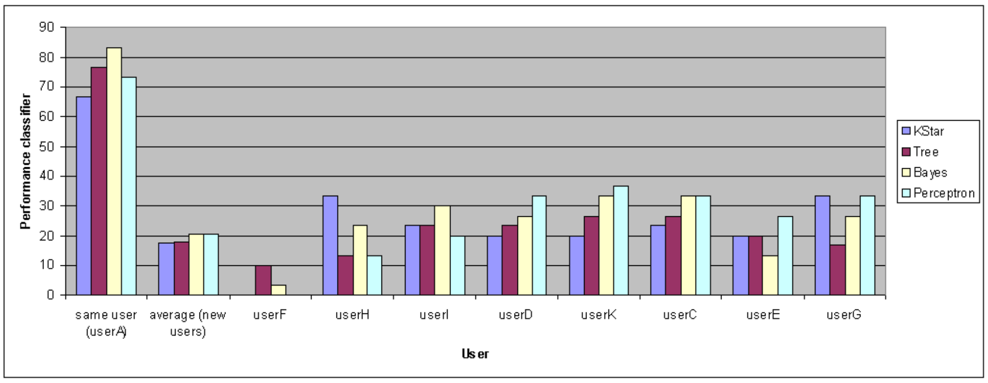


Figure 6.6: Performance of the classifiers trained on user A's data, tested on other users. Note: Users ordered on entropy score of their task distribution – low to high.

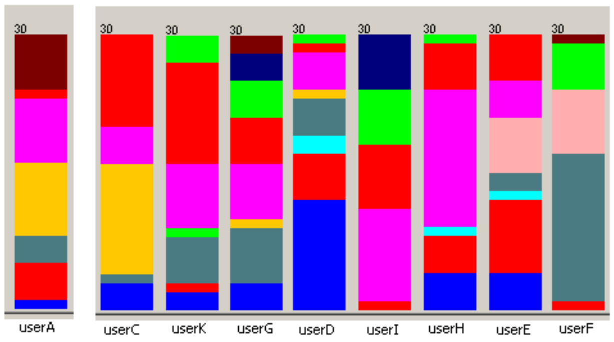


Figure 6.7: Task label distributions, users ordered on classification performance with user A's classifier.

In figure 6.6 you see that although the classifier works fine on user A's test data it reaches a performance of only maximally 30% on other users. It differs per user which classifier performs best, there is no clear advantage for one of the classifiers when it comes to generalizability. We pictured the task distributions of the users in the order of classifier performance average in figure 6.7. You see a higher performance can be reached for users whose task distribution is more similar to the task distribution of user A. For user F the task distribution differs very much from user A's, so as expected the trained classifier does not work at all. Averaged over all users the classifiers all perform around 20% which is below the baseline given by ZeroR (see figure 6.1). We can conclude from this that a classifier trained on one user does not work on other users data.

Our next question was whether a classifier became 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. In a leave-one-out manner we took 30 instances of all but one user and trained a classifier upon it. Then we tested its performance on the left out user's to test the generalizability of the model. In figure 6.8 it becomes clear that dependent on the user a particular classifier seems to have best generalization properties. For user I, D, K and C Bayes clearly works best, for user F and H the Decision Tree is best and for user E KStar clearly wins. Averaged over all users Bayes is best in generalization, followed by Decision Tree classification and KStar. In general a maximal performance of about 40 to 50% can be reached on new users in this way. Nevertheless the average classification performance is only 20 to 30%. This is better compared to training on one user's data, but far from satisfactionary.

We can conclude from this analysis that a classifier trained on a mix of users' data does not generalize well to new users.

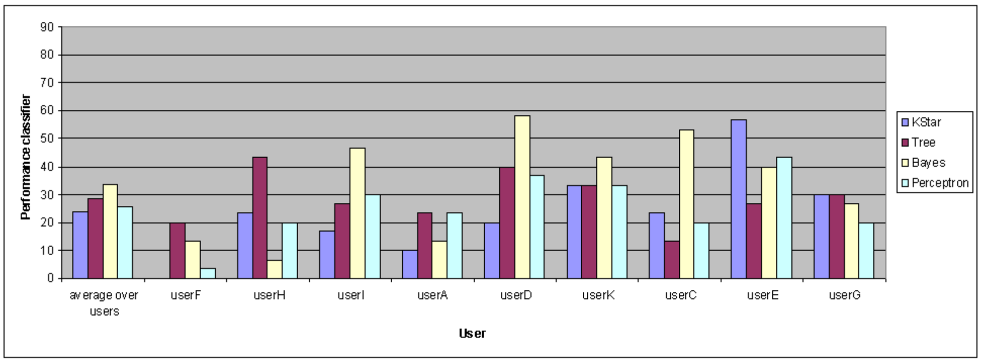


Figure 6.8: Performance of the classifiers trained on all-but-the-tested user's data. Note: Users ordered on entropy score of their task distribution – low to high

6.2.3 Conclusions about individual differences

Recognizing tasks on basis of PC activity seems to be a very personal issue. 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 their own individual manner of working. Besides the factors analyzed here, other factors might be of influence too. Users might for example differ in their interpretation of what makes up a specific task and in how precise they label their activities. Within one user however there is a general structure in activities which makes task recognition possible.

Results show that a classifier can best be trained for one particular user. Initializing a classifier on other users' data seems not a very useful approach. A good way for starting to classify a new users behavior seems sampling a representative set of his tasks. A dataset as small as 2.5 hours of data seems large enough to train a classifier yielding reasonable task recognition performance.

6.3 Remarks on our framework of task recognition

In chapter 4 our general framework for task recognition was presented with a set of features, a set of tasks labels and the idea of classifiers mapping from the features to our labels. More general questions regarding this task recognition framework are:

- Which features turned out to be most useful?
- Which tasks were typically confused and what does that mean for our task labels?
- Can classification be improved by giving the classifier information about the previous time frame?

In the next sections we describe our further analysis to answer these questions.

6.3.1 Most useful features

To analyze which features are most useful for task recognition we used Weka to compute the information gain ratio of our features. First of all Weka was provided with one big dataset containing 30 sampled data points of each user. In figure 6.9 all features with an information gain ratio above zero are plotted. It can be seen that taking the data of all users together the most informative features are specific programs like time spent in Powerpoint, Excel or program applications. This is as expected as these are clearly coupled to tasks (presentation making, analyzing data and programming respectively). These coupling will be true for each user. Besides information about used applications the user itself appears as useful feature. This means that knowing which user is acting helps in predicting the task, which shows that task prediction is user specific. There seems to be no useful information in other features like amount of clicking or typing or time of the day or task switching when the data of all users is mixed up.

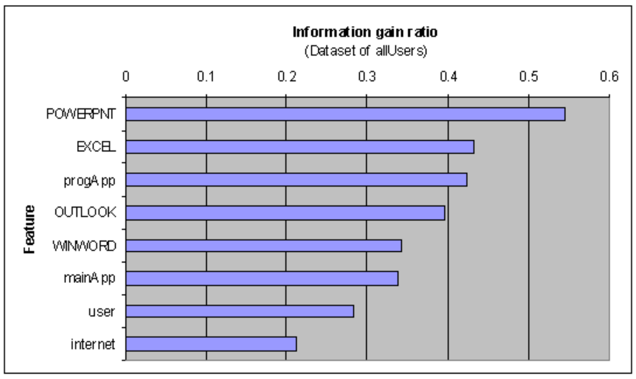


Figure 6.9: Information gain ratio of the features, given one dataset containing data of all users.

Our intuition was that for each user taken apart other features could turn out to be useful. So in the second feature analysis Weka was provided with datasets of 30 sampled data points of one user at a time. For different users different feature turned out to be most indicative. In figure 6.10 we averaged the calculated information gain ratios of all separate users to give some overview. In general the main application that was used is very indicative for all users. Besides the amount of time spent in specific applications like Powerpoint or Excel also characters, spaces, backspaces, special keys and scrolls seem to be useful to discriminate tasks now. So taking a person separately there is structure in his or her mouse and keyboard activity that can be used for discriminating tasks. Also features like daytime, time spent in the most prominent application, number of used applications or number of switches now seem informative. Given one user his style of working can help the classifier in discriminating tasks. The fact that all these features are not useful when the data of all users is mixed up again shows that there is no general structure in behavior among users. Each users behavior is very individual.

From this analysis we can conclude that the main application and time spent in specific applications are very robust features that hold among all users. Besides that looking on a per user basis features about clicking or typing behavior, as well as time of the day or switching behavior are useful for task recognition.

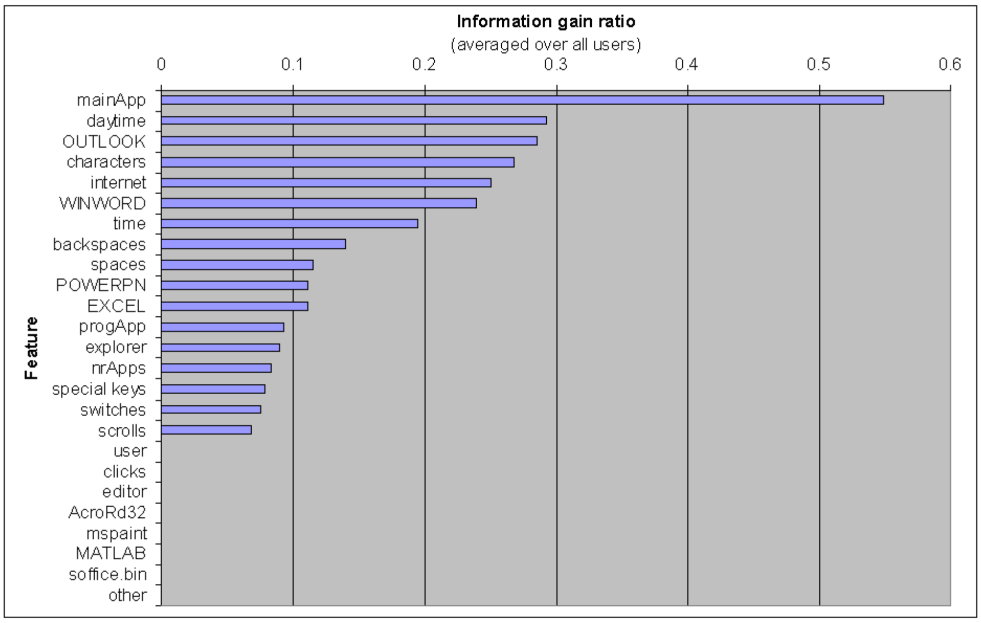


Figure 6.10: Information gain ratio of the features, given separate datasets for each user.

6.3.2 Analysis of confused task labels

We were interested in which task labels got confused in order to validate our set of defined task labels. In figure 6.11 the confusion matrices for the different classifiers are shown.

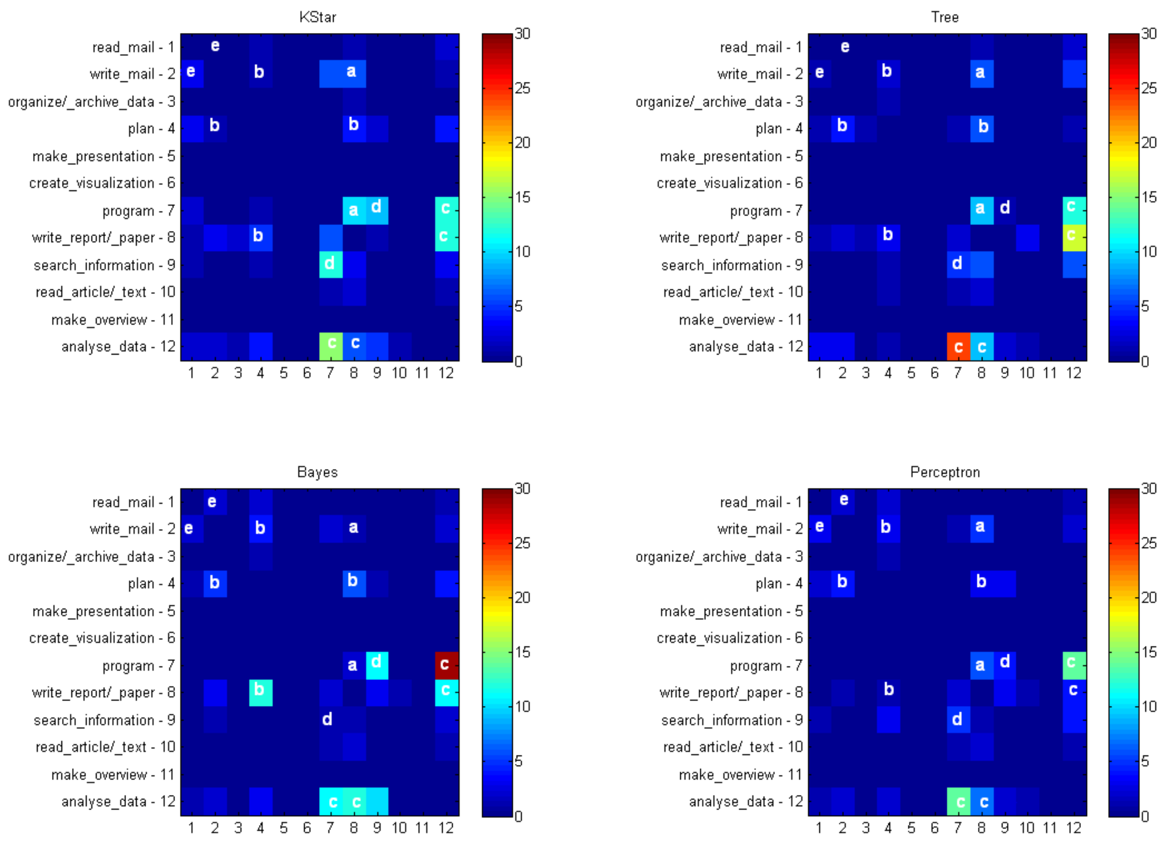


Figure 6.11: Confusion matrices of the classifiers trained on user A's data. Note: only errors depicted, diagonal is set to zero.

In general all classifiers show similar error patterns. First of all there are some mislabelings due to the similarity of the tasks. The tasks 'write mail' and 'program' are sometimes recognized as 'write report' which is understandable as they all involve much typing. Moreover the task 'plan' seems difficult to recognize. It is often confused with mail or report writing (b). This confusion is understandable as planning involves using the outlook calendar and typing in Word to make to-do lists, which makes the pattern of activity similar to mail or report writing.

Besides some mislabelings due to the similarity of the tasks another more important kind of error becomes clear. Tasks that the user combines are often confused. For example 'analyze data' is often confused with 'programming' and 'write report/ paper' (c). The classifier simply classifies 5 minute frames. In analyzing data the user might from time to time switch to some programming to make plots in Matlab or to report writing by noting down observations or conclusions. So these errors might actually be not the fault of the classifier, but they might be due to the labeling of the user, who is unaware of these switches and labels all activity as analyzing data. A similar pattern of confusion is observed for programming and searching information (d). Also in this case the tasks themselves are not very similar, but they are simply often combined by the user. While programming the user switches to searching behavior when he or she looks on the Internet for solutions.

A final observation is that ‘read mail’ and ‘write mail’ are confused (e). These are both tasks that have a very short duration and the user might switch between mail reading and writing within one time frame of 5 minutes. Therefore one could propose to better merge these task labels as ‘process mail’.

In general one can say that all classifiers are prone to these error types. The Perceptron can minimize them, but still has the same problems. We can conclude that the errors are not due to the models not finding the right structure in the data, but the data itself is fuzzy. There are not always clear boundaries between the tasks. To some extent this is inherent in the tasks as certain tasks look very similar. To an even larger extent this is due to the users’ manner of labeling of their interleaving activities.

We can conclude from this analysis that the classifiers mainly confuse intervening tasks due to the labeling of the user. The user seems to think more in terms of general goals he or she is trying to accomplish. The computer recognizes the method the user is adopting. There is thus a slight mismatch in labeling. To solve this problem one might on the one hand try to let the user accept a more fine grained labeling where switches to subtasks and intervening tasks are differentiated. On the other hand one might wish to interpret the tasks that the computer recognizes further and classify patterns of subtasks as main task the user is accomplishing.

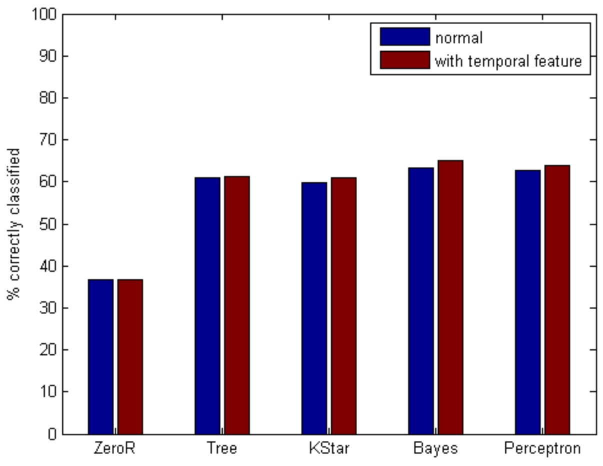


Figure 6.12: Performance of the classifiers with and without the feature ‘task in previous time frame’, averaged over users.

6.3.3 Effect of adding a temporal feature

A last question we want to answer is whether giving the classifier information about the previous time frame can improve classification performance. Our basic model just uses the information of the 5 minute time frame to classify the activity. Nevertheless information from the previous time frame may also be useful for task recognition. As simple temporal feature we added the task of the previous time frame. In cases the user had provided a correct label this was used, in other cases the label assigned by the classifier was used.

In figure 6.12 it can be seen that averaged over all users there is not much improvement for adding our temporal feature. Taking a look at each user separately gives another view. In figure 6.13 you see that for user A and C all classifiers improve much with our temporal feature, about 5 to 10 % points. For user E, G and K there is not much difference. For user D, F, H and I it depends on the specific classifier whether classification is better or even worse when taking the task of the previous time frame into account. A possible explanation is that the users differ in work style. For example user A and C might work very focused, without much switching, so that the previous task is indicative of the current task. For a user switching a lot between tasks information about the previous time frame might often be confusing for the classifier.

We can conclude from this analysis that one cannot say in general that adding information about the previous time frame shows significant improvement of classification performance. Dependent on the user and classifier it can boost performance, but sometimes also worsen it. In further research it should be investigated how information over time could best be integrated.

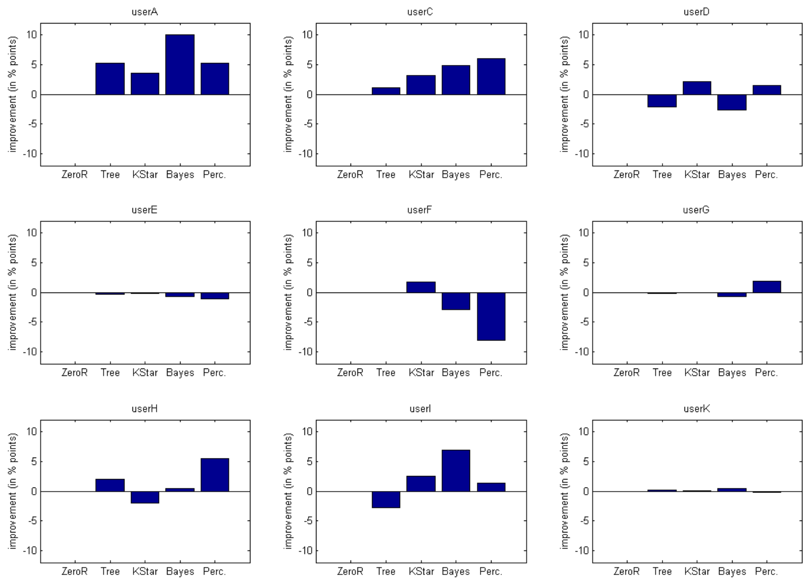


Figure 6.13: Performance of the classifiers with and without the feature ‘task in previous time frame’, per user.

6.3.4 Conclusions about our framework of task recognition

Regarding our chosen features all sorts of features turned out to be useful. Information about applications is a robust feature among users, whereas mouse and keyboard features as well as work style features are useful to discriminate tasks on a per user basis. No revisions regarding the features have to be made.

Also our chosen set of task labels seems appropriate in general. What turned out is that some tasks involve other tasks as subtasks. Moreover using information of the previous time frame was shown to boost classification performance for some users.

Combining the last both insights leads to the proposal of a better framework depicted in figure 6.14. A simple classifier could be used to analyze 5 minute time frames, which yields a sequence of event labels. On a higher level several event labels could together constitute one task. For example the combination of writing some text and from time to time looking for information could together constitute the main task of writing a report. By taking this mix of events over time into account a second classifier could recognize the main task the user is performing based on a series of event labels. In this way the automatically assigned task labels would probably better match the labels humans intuitively use to describe their performed activities.

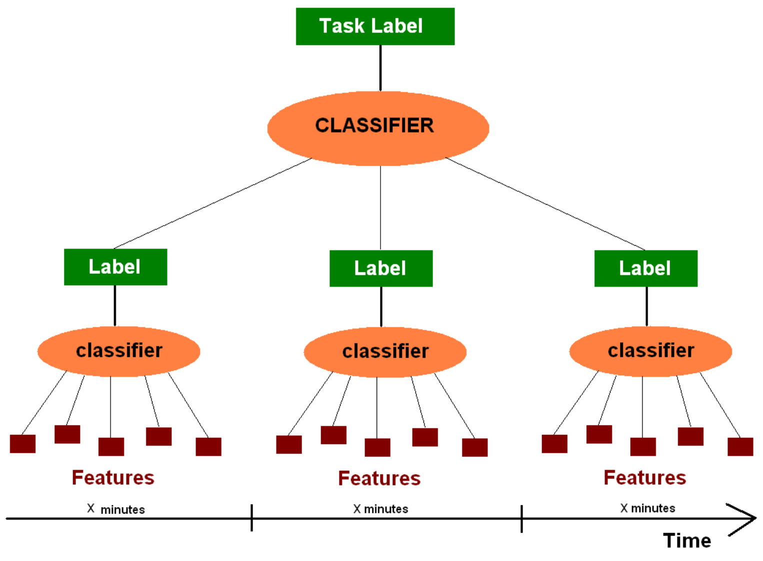


Figure 6.14: Proposed framework taking sequences of activity into account.

Chapter 7

Discussion

In this thesis we presented work on task recognition based on PC activities. Our first research question was on which level of abstraction humans would describe their own PC activities. On basis of a questionnaire a set of typical knowledge worker tasks was identified. Our next research question was how knowledge workers work at their PC. A set of low level activity features and a framework for task recognition were chosen, which were shown to be useful in a pilot study.

Our main research question was whether it is possible to recognize the defined tasks on basis of the captured PC activities. Our research has shown that task recognition on basis of PC activity is challenging but feasible. It involves more than a simple one-to-one mapping between an application and a task due to interleaved activities, switches to subtasks and a mix of applications used. Comparison of several classifiers revealed that there is not one classifier that clearly works best in this domain. Task recognition is very personal. Different users have different work styles and task mixes. Moreover users might differ in their interpretation of what makes up a specific task and in how precise they label their activities. Nevertheless we saw that on an individual basis the classifier learns to recognize tasks quite fast, yielding a performance up to 80% which is reasonable high, considering 12 possible tasks labels. Opposed to other research, in which clearly structured tasks are modeled (see e.g. Natarajan et al., 2008), this work has shown that task recognition also works for less structured tasks and more spontaneous activity.

Important observations

Several important observations regarding our research method were made. First of all getting a good ground truth labeling of users' tasks turned out to be rather difficult. A tool suggesting task labels made the labeling task somewhat easier for the users, but still annotating stays difficult as it cost the users time and effort. For some users the amount of checked data was little, but nevertheless enough for training classifiers in our analysis and gaining insights. Furthermore, some users reported that it was difficult to remember what one has exactly been doing. Especially when some time had passed or the user had been interrupted and switched tasks, it was difficult to estimate in which time frame what has been done. We instructed the users to only confirm the labels when they were certain about the performed task, but nevertheless the collected data may contain some human mislabellings. A positive observation regarding data annotation is that the users are curios and interested in whether the tool comes up with correct labels, especially in the beginning, which motivates them to regularly check the labels. This curiosity and interest could be further exploited by for example giving points for correct guesses or reward the user for checking labels, making the annotation and the tool in general fun to use and game like.

A further important observation we made in our experiments is that humans think in terms of broader goals, not in terms of the specific methods used. This is in line with the differentiation that Heinze (2003) makes in Tahboub (2006), describing an intentional level and activity level. One might see the more detailed description the task recognizer comes up with as describing the specific activities performed, including all subtasks. Users agree that they actually performed these subtasks, but they themselves describe their tasks performed during a day on a less detailed level, labeling only their intended main tasks. Taking this further, users might label the tasks performed during a week or month on an even higher, lesser detailed level of description, like for example 'this week I worked on project X'. These task description levels of humans should be considered when choosing a suitable level of description for recognizing knowledge workers' task.

A final insight gained in this study is that being monitored can already influence the work style of a user, which confirms the results of Johnson and White (1971). Some users indicated that using the task recognition tool created awareness about switching and away from keyboard behavior. When the tool was on they described to work more focused. In general the users indicated that they quickly got used to the presence of the tool. By some users it was reported that in the beginning they got a feeling of needing to work hard and account for all their performed activities. This could be due to the fact that in our study the users needed to constantly check their task labels. This is not the case for the real application. Nevertheless one should take into account that the tool could have the effect of pushing the knowledge worker to perform. On the one hand this is good as it causes focused activity and high productivity. On the other hand a feeling of stress should be prohibited and the importance of breaks be highlighted.

Implications of our results

Our results have different practical implications for a tool supporting the knowledge worker. The most important observation is that different users show a different pattern of behavior when performing a task. As a consequence the classification model should be trained on one specific user to yield optimal task recognition performance. When the tool is applied to a new user we face the so called cold start problem: without any data to learn from, the classifier will not be able to able to classify correctly. A solution would be to let the tool ask the user what he or she is doing at several times during one week, collecting a representative set of annotated data. As we saw as little as 2.5 hours (30 instances) of representative training examples is enough to train a good model. After this week the tool can then quite reliably start to recognize this user's tasks. An alternative approach would be to pretrain a model on other users' data and then use active learning to improve this model. When the recognition module is quite sure about the task that this user is performing it could propose a label to be checked. In this way the recognition module could be used right away and trust from the user in the tool could be gained. After some classifications with high certainty the tool could start asking for task labels of tasks it is still unsure about as this will cause main improvements to the classifier. Finally the classifier will learn all behavior of this user and can in the end 'forget' the training examples of other users.

Another result having implications for the tool is the found variation in task mixes and behavior. Also within one user the task mix or the behavior might change over time or the user might start performing new tasks. The system should keep adapting to this. It is therefore important to regularly update or retrain the model. It would also be good for the flexibility and applicability of the tool when users could add new task labels to the recognition module when required.

Remaining challenges

The biggest improvement regarding the task recognition performance could be gained by exactly matching the task labels that users intuitively used. We saw that some tasks involve other tasks as subtasks. A series of these subtasks over time could be used to label the activity as the main task the user performed. Temporal models like Markov models or conditional random fields could be considered for modeling these sequences, like is done in related research (e.g. Oliver, Garg, & Horvitz, 2004; Duong et al., 2005; Natarajan et al., 2008). Moreover one might 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.

Another remaining challenge is that some users indicated that not all their work can be captured on basis of PC activity, as they spend time in meetings or phone calls. To be able to capture this one might wish to extend the system by adding information from the user's calendar or mobile phone.

In general, the collected data about the performed tasks and working style of knowledge workers is itself very valuable. On basis of this user behavior data one could further study their way of working and typical patterns of behavior. This could help improving the system by better adjusting the task recognition framework to the domain. Moreover possible points on which users could be coached could be identified.

Using task recognition in a feedback tool

Our results have shown that knowledge workers' tasks could be reliably recognized. The few 'errors' still made are often merely finer grained labellings. These tasks could thus be presented to the user as a mirror of activities, which increases the possibility for self evaluation (Bandura, 1991). Furthermore one could add an indication of how demanding work was by linking each task to a score. Writing a report could for example be rated as more demanding than reading something. Besides the task also information about the users' behavior is readily available, as for example the number of characters or backspaces typed, which one might visualize for more insight about for example ones productivity or level of concentration. Patterns of work demand, productivity or concentration could then be visualized over the day, week or month. The curves could be shown in relation to past performance or a personal standard (Manz & Sims, 1980). Performance of different colleagues could also be compared. However one has to note that this social comparison can have positive or negative effects (Bandura, 1991).

We can conclude that on basis of our task recognition module automatically high level overviews over performed task can be made, which is an improvement compared to the already existing systems (like SLife or RescueTime).

Things to consider

When creating a good feedback tool effort should be spent on the visualization used. Users indicated that they would like to gain insights in one glance. One could think about a time bar showing the tasks and amount of switching between them or a score indicating how focused someone has been working. By valuing the observed behavior the user can be pointed toward bad habits to get rid of or toward good patterns of behavior to sustain in.

Another important aspect to consider is the user interface. For the tool being accepted and used by knowledge workers it should be very user friendly and have a good user interaction. It should not be too obtrusive, annoying the user with pop-ups or demanding much effort. As our study revealed users easily switch off the tool when they feel busy and the tool demands effort, slows down the computer or annoys them. Certainly many useful tips for a successful tool could be gained by considering the paper by Swartz and Nardi (2003) about why people hate Words' paperclip. The system is used during working, so it is very important to let the user have sufficient control over the system. Especially when the user is very busy it should be possible to keep the tool in the background. Moreover it is important for the system to gain trust of the user. When the user notices that the tool does not work correctly and keeps recognizing wrong tasks this can easily frustrate him, as our study revealed. By starting with presenting tasks that are recognized very certain trust in the system could be built.

The most important aspect that has to be considered when further developing a tool supporting knowledge workers is privacy. In our study the users had no privacy concerns. Most data was locally stored and things stored in a common database were anonymized and all letters turned to X to prohibit reconstruction of contents. We chose to use one common database to enable that logging data from different PCs used by a knowledge worker could be integrated. An experienced disadvantage was that the computers needed to have a connection to the Internet and communicate with the database. One might prefer storing all data only locally. In that case also content information might be exploited for the tool, without privacy harms.

Further possibilities

With task recognition as a basis the tool supporting knowledge workers could be further extended with several functionalities. One could for example add content information to gain more insights about the context of a task or the specific project it was performed for. Moreover, given the typical behavior of the user for a task one could detect deviating behavior, when someone for example is anomalously often using the backspace key or switching windows very often. This could be a simple indication of stress that could be used to warn the user. Finally a digital coach application could be possible, helping knowledge workers to improve their work style and diminish stress. The digital coach could attend the user to interesting patterns of behavior, create awareness and insights, help the user improve his behavior, give tips and guidance and assist him. He might suggest users with a comparable work style or task mix, in order to facilitate helping each other or giving each other tips. We conclude that in general many applications supporting the knowledge worker and ensuring his well being are possible.

Chapter 8

Conclusion

This thesis aimed at designing a system that can recognize and log the tasks a knowledge worker is performing at the PC. The characteristics and demands of this specific domain for task recognition were analyzed. The knowledge worker typically experiences many interruptions and task switches and his task performance is much influenced by outer influences and often involves incidental switches between different applications. The tasks to be modeled are thus less structured than tasks in comparable research. A framework for task recognition was chosen, in which a classifier uses low level PC activity collected over a five minute time frame to get a relatively stable account of task related activities.

A set of suitable task labels was determined based on a questionnaire. Answers revealed that humans do not think in terms of applications, but that the mentioned tasks often involve the use of a specific mix of applications. Therefore task recognition seems not to be a simple one-to-one mapping between an application and a task. Using a logging tool various PC activities were captured and several features were extracted, like number of clicks, scrolls, typed characters or backspaces, as well as time of the day or number of window switches. The appropriateness of these chosen features was confirmed in an initial pilot study. Four different classification approaches were chosen for implementing task recognition: a nearest neighbor, rule based, Naive Bayes and neural network approach, which all base their classification on different principles.

In a field study, data of nine knowledge workers was collected in order to test which of our chosen classifiers reached best task recognition performance and learned fastest. Results showed that after only 2.5 hours of training data (30 instances) a task recognition performance of 80% accuracy could be reached. There was however no clear advantage for one of the classifiers with a high variance in performance between users. Due to considerations of processing speed a simpler classifier like a decision tree or Naive Bayes could best be chosen for fast and efficient task recognition in this domain. Deeper analysis showed that users differ in terms of the task mix they perform and that a mix with a high entropy is in general more difficult to classify. Moreover users show a very individual pattern of activity. A classifier model trained on other users' data is not suitable for classifying a new users data, which means that a task recognizer needs a personal user model.

In general the most indicative features are the used applications. On an individual basis also typing and clicking behavior as well as time of the day or amount of switching help to discriminate tasks for a specific user. Error analysis revealed that all classifiers suffer from the same errors which indicates that the problem is not that the different approaches fail to find the structure in the data, but that in general the labeled data is fuzzy. Most errors turned out to be due to inconsistent labellings of the users, who tend to label intervening subtasks as the main task they are performed for. Adding information of the previously performed task could in some cases improve task recognition performance. These findings indicate that considering a sequence of tasks over time could help in giving task labels that humans intuitively use.

Finally, the possibility to automatically recognize the tasks a user is performing enables the creation of overviews over the day or week, to give the user more awareness and first insights about his or her working process. In further research we will investigate how the basic task recognition module can be further extended, realizing our final aim of a support tool for knowledge workers, which can help them to improve their behavior toward more productivity and a higher well-being.

References

Albrecht, D. W., Zukerman, I., Nicholson, A. E., & Bud, A. (1997). User modeling: Proceedings of the sixth international conference, um97. In C. P. Anthony Jameson & C. Tasso (Eds.), (p. 365-376). Springer Wien New York.

Baker, C. L., Saxe, R., & Tenenbaum, J. B. (2009). Action understanding as inverse planning. *Cognition*, *113*(3), 329 - 349.

Bandura, A. (1991). Social cognitive theory of self-regulation. *Organizational Behavior and Human Decision Processes*, *50*(2), 248 - 287.

Bishop, C. M. (2006). *Pattern recognition and machine learning.* Springer.

Blokpoel, M., Kwisthout, J., & van der Weide, T. (2010). *How action understanding can be rational, bayesian and tractable.* (Manuscript under review)

Cheyer, A., Park, J., & Giuli, R. (2006). *Iris: Integrate, relate. infer. share.* (Tech. Rep.). Defense Technical Information Center OAI-PMH Repository [http://stinet.dtic.mil/oai/oai] (United States).

Cleary, J. G., & Trigg, L. E. (1995). K*: An instance-based learner using an entropic distance measure. In *Machine learning: Proceedings of the twelvth international conference.*

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* (pp. 175–182). New York, NY, USA: ACM.

Dragunov, A. N., Dietterich, T. G., Johnsrude, K., McLaughlin, M., Li, L., & Herlocker, J. L. (2005). Tasktracer: a desktop environment to support multi-tasking knowledge workers. In *Iui '05: Proceedings of the 10th international conference on intelligent user interfaces* (pp. 75–82). New York, NY, USA: ACM.

Duong, T., Bui, H., Phung, D., & Venkatesh, S. (2005, jun.). Activity recognition and abnormality detection with the switching hidden semi-markov model. In (Vol. 1, p. 838 - 845 vol. 1).

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, November). The weka data mining software: an update. *SIGKDD Explor. Newsl.*, *11*, 10–18.

Horvitz, E., Breese, J., Heckerman, D., Hovel, D., & Rommelse, D. (1998, July). The lumiere project: Bayesian user modeling for inferring the goals and needs of software users. In *Fourteenth conference on uncertainty in artificial intelligence* (p. 256-265). Morgan Kaufmann Publishers.

John, G. H., & Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the eleventh conference on uncertainty in artificial intelligence.*

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

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.*

Manz, C. C., & Sims, J., Henry P. (1980). Self-management as a substitute for leadership: A social learning theory perspective. *The Academy of Management Review*, *5*(3), pp. 361-367. Available from `http://www.jstor.org/stable/257111`

Morrison, J. B., Pirolli, P., & Card, S. K. (2001). A taxonomic analysis of what world wide web activities significantly impact people's decisions and actions. In *Chi '01: Chi '01 extended abstracts on human factors in computing systems* (pp. 163–164). New York, NY, USA: ACM.

Myers, K., Berry, P., Blythe, J., Conleyn, K., Gervasio, M., McGuinness, D., et al. (2007). An intelligent personal assistant for task and time management. *AI Magazine*, *28*(2), 47-61.

Nardi, B. A. (1996). Context and consciousness: Activity theory and human computer interaction. In B. A. Nardi (Ed.), (p. 69-101). Cambridge, MA: MIT Press.

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

Oliver, N., Garg, A., & Horvitz, E. (2004). Layered representations for learning and inferring office activity from multiple sensory channels. *Computer Vision and Image Understanding*, *96*(2), 163 - 180.

Oliver, N., Smith, G., Thakkar, C., & Surendran, A. C. (2006). Swish: semantic analysis of window titles and switching history. In *Iui '06: Proceedings of the 11th international conference on intelligent user interfaces* (pp. 194–201). New York, NY, USA: ACM.

Quinlan, R. (1993). *C4.5: Programs for machine learning.* San Mateo, CA: Morgan Kaufmann Publishers.

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* (pp. 1–10). New York, NY, USA: ACM.

Rath, A. S., Weber, N., Kröll, M., Granitzera, M., Dietzel, O., & Lindstaedt, S. N. (2008). Context-aware knowledge services. In *Workshop on personal information management, chi '08.*

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. *J Appl Behav Anal.*, *21*(4), 401409.

Shen, J., Li, L., Dietterich, T. G., & Herlocker, J. L. (2006). 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: ACM.

Swartz, L., & Nardi, B. A. (2003). *Why people hate the paperclip: Labels, appearance, behavior, and social responses to user interface agents* (Tech. Rep.).

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

Appendix A

Questionnaire

see 'Questionnaire.pdf'

A.1 Results of the Questionnaire

see 'Results of the Questionnaire.pdf'

Appendix B

Intro Well Working

One month after starting my internship, TNO started a so called 'kennis project' (knowledge project) on the topic of well working. The well working project is aimed at giving the knowledge worker more awareness about his functioning in order to lead him to a state of well working. As the topic of my internship is narrowly related to this I became part of the project team.

B.1 Aim

As already explained, knowledge workers are typically confronted with great amounts of information, are often interrupted and often work reactively. This way of working can easily result in stress and decreased efficiency. The idea behind the well working project is to develop a tool that can monitor and aid a knowledge worker in his self management to enable him to reach a pleasant state of well being in his work. To reach this goal three aspects are addressed, which will be described in the following.

Domain knowledge about measuring work quality

First of all theoretical knowledge on well working is gained. On basis of a literature study a definition of well working is formulated. Several models are considered to investigate what the determinants of well working or stress are and what its consequences are. Moreover an overview is made of how the determinants as well as the consequences of well working or stress can be measured.

Technology for activity registration

The self management tool is based on sensor data, i.e. logs of PC actions. Therefore it is investigated in how far stress and fatigue can be deduced from this sensor data. What are the possibilities, what are good features to detect stress, fatigue, well working or the efficiency of a knowledge worker? A tool that logs user activity is used together with a tool to annotate activities with task labels and indications of well being. In this way a test set of annotated logging data is created.

Requirements digital coach

In its simplest form the detected information is just presented to the knowledge worker to give him insights. An advanced form is using the detected information in a digital coaching system. It is important to investigate the needs the knowledge workers have and the resulting requirements for the software. Specifically the acceptation and success criteria for such digital coaches should be investigated.

B.2 Goals

Different knowledge is available within the organization TNO: The department 'Arbeid' knows much about the way and context in which people work. The department ICT knows a lot about using sensor data and aggregating it towards useful information and the department Human Factors has much knowledge on human computer interaction. The knowledge from these various fields of expertise will be combined in this knowledge project to yield new insights. More general goals of the project are finding commercial partners and formulating focused research questions.

Appendix C

Well working model

A literature study was performed to gain more insight in the context of well working.

I participated in this part of the project by reading some of the articles and filling in information from the articles in an overview table. Together with my colleague Pepijn Vos I wrote down the most important conclusions to present to the other team members. Moreover I participated in a conference call with the department 'Kwaliteit van Leven' and read the feedback they gave on our ideas. Finally I shortly presented the status of our sub-team to the rest of the team and prepared an overview of determinants and how they could be measured on the PC for further discussion with the group.

Definition

First of all a definition for the Dutch term 'arbeidsvreugde' was sought. Terms typically found in the literature were the positive terms 'well being', 'employee/ job satisfaction', 'engagement' or 'commitment' and the negative terms 'stress' or 'burn out' (Figure C.1).

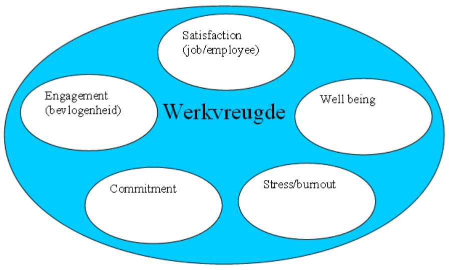


Figure C.1: The terms around the concept of 'werkvreugde' (well being at work)

Effects

Moreover the effects of 'werkvreugde' (well being at work) were investigated. In general one can say that well being has many positive effects on the human and the organization and therefore should be facilitated.

Determinants

The fact that the well being of an individual can be influenced and regulated led us to the question: What are the most important determinants of well being at work? Analysis of various models from the literature showed that various aspects play a role: The working conditions, the work itself, personal characteristics and private aspects (Figure C.2).

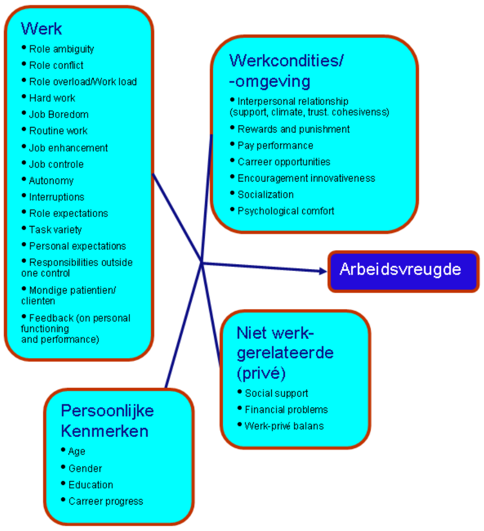


Figure C.2: The different types of determinants for well being at work

From the huge set of factors influencing well being at work the nine most important determinants were exposed:

- Sociaal klimaat
- Uitdagende werkinhoud
- Interrupties
- Nut
- Behalen van resultaten
- Versnipperd werk
- Zelfstandigheid
- Fysieke werkomgeving
- Werkdruk

Measuring

Until now assessment of the employees working context and psychological state is done by means of questionnaires. The intention of our tool is to measure the working context and the state of the knowledge worker by means of interpreting logs of pc activity. Therefore it was investigated which determinants could effectively be measured with logging data. The whole list of determinants was divided into things to ask the user once, like his age or gender, things to ask the user from time to time because their might be changes, like his role, and things that can be indeed monitored with the pc. From the list of things to be monitored with the pc an initial set of easy to measure things was selected to focus on in the remainder of this project:

- *Interruptions*, by means of how frequently the user accesses the Outlook Inbox or the frequency he is away from keyboard
- *Task Switching*, by means of the frequency with which the user changes applications
- *Task variety*, by means of the type of programs used or the detected tasks
- *Focus*, by means of how long the same application is used
- *Stress*, by means of the frequency of backspaces or deviations from a normal typing pattern

Appendix D

Workshop

I participated in this part of the project by meeting with two other colleagues, Pepijn Vos and Sonoko Takahashi. In the first session we discussed what topics we would like to address in the workshop in order to gain useful insights from knowledge workers. In the second meeting we discussed the self management model and prepared the workshop in more detail. My colleagues prepared the presentation, on which I gave feedback. During the workshop I participated in the sessions and took notes. Finally I worked out the results of the workshop.

In order to get a clearer picture of the demands for the software a workshop with members of the well working team was performed. Seven people participated in this two hour workshop. Four aspects were addressed:

1. What should the self management tool be able do?

2. How should the interface and the visualizations of the tool look like?

3. When and how is the self management tool used?

4. Which aspects have to be taken in mind when introducing the tool?

The idea was to imagine the optimal, perfect tool, without thinking about technical possibilities and limitations. In the following the course of the workshop and the results of the four sessions will be presented.

Introduction

As analysis has shown, supporting knowledge workers in their self management is an important aspect of the tool. In the introduction of the workshop therefore the process of self management was shortly explained:

Self management is a process in which the individual himself *observes* his work and *judges* it against established norms. On basis of this the individual eventually *changes* his way of working to reach his *norms* (Figure D.1). He may also evaluate and eventually adapts his norms.

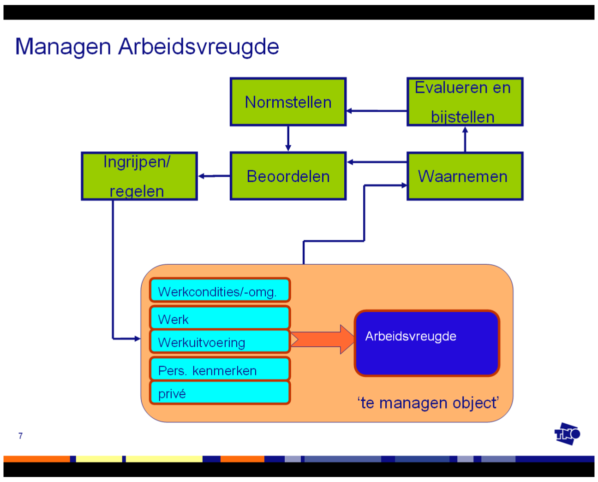


Figure D.1: Working environment of a knowledge worker

Session 1 – What should the self management tool be able do?

In the first session the participants were asked to think about the functionalities the tool should have to optimally assist them in their self management.

Preceding to the workshop interviews had taken place with people who dropped out of work because of burnout. These interviews gave an initial picture of the functionalities the software should have to optimally assist workers. These ideas were shortly presented to the workshop participants. The participants were then asked to write down additional ideas, which were later discussed in the group. In the following the supporting functionalities per step in the self management cycle are discussed.

Observe

First of all the tool can help by making the observation of your work easier. In general participants indicated that the self management tool should help to gain insights and discover aspects you were unaware of before. Several factors could be measured and presented. The tool could for example present overviews of the activities performed on a day, which could give information about the fulfillment of tasks as well as insights into the amount of time spent on specific things like email. Moreover the system could provide overviews over the documents worked on, the time spent on them and statistics about text input like the average typing speed to give an indication of how well working went (flow). Moreover someone wished to gain insights into with which people he often works and how this is related to his well working.

Judge

The tool can also help in making good judgments about ones work. The interviews with people who had experienced burn out led to the following ideas. The system could help to judge in how far the current situation deviates from the desired. It could signal things and give positive and negative feedback to the user. For example the knowledge worker could be warned when his stress level rises or when his working pattern deviates and might indicate a bad way of working. However it turned out that it is important to not only focus on things going wrong, but also giving the user positive feedback or compliments.

The participants of the workshop further added the following things. The tool could help them to track whether they indeed spent time on tasks with high priority. It could assist in updating to-do lists and yield motivating 'tsjakkaa' moments when tasks can be crossed out. Moreover it could contrast your planned time or set deadlines with the actual time of completion, which can in the future help to make better time estimates and plannings. The system could also present trends, for example the development of your stress curve to show you where you are heading at, to prevent reaching a state of burnout. Finally the tool could give you insights in what kind of working days typically lead to a state of high satisfaction and what kind of days or working styles not.

Change

Instead of simply monitoring you or helping you to judge your style of working the tool could also help you to change your behavior. On basis of the interviews two types of actions became clear. On the one hand the system could merely give tips or advises. On the other hand it could directly interfere by for example blocking things on your computer. In the workshop some more details were filled in. The system could for example give warnings when the user shows a high amount of task switching, when the software detects that planning problems are arising or the performed work is too monotonous or boring. Professional feedback, tips or best practices of others were appreciated. These tips could for example indicate how to diminish the stressors.

As possibilities to let the computer infer the participants mentioned that the tool could automatically block the calendar when the workload is too heavy or it could filter incoming emails according to their importance. Another interesting functionality of the tool could be to protect the user's flow, i.e. when a status of flow is detected, possible disturbing factors like pop-ups of incoming emails become blocked.

Norms

The tool could also be used to establish and evaluate norms. On the one hand the tool could ask the user to explicate his norms. The user could set goals for himself, for example to work less hours per week or to empty his inbox regularly, or indicate which activities have priority for him. On the other hand the tool itself could learn what good norms are. A benchmark could be established on basis of the data of other users. The interviews revealed that users would like to bring to light whether ones own norms are realistic or need to be adapted. This could be done by comparing oneself with the benchmark.

Session 2 – How should the interface and the visualizations of the tool look like?

In the second session we asked the participants to think about the interface and the visualizations of the self management tool. All members were asked to draw their ideas and afterwards discuss their sketches with their neighbor and write down the concept behind their visualizations. Finally all ideas were shared with the group.

Again, first the ideas of the interviewed people were presented to the workshop participants. These people mentioned two different forms of visualizations. Very technical visualizations included bar charts and graphics depicting trends over time. More metaphorical visualizations included using a traffic light to depict your stress level or a balance meter to depict whether you are out of balance. Another visualization mentioned was using a character that speaks to you or moves in a way that depicts your state (e.g. hanging shoulders or full of energy).

As it turned out, the participants of the workshop drew the same two types of visualizations: On the one hand very technical visualizations showing the facts in detail, on the other hand simple visualizations to show you your state at one glance. A participant explicitly noted that in the first place simple visualizations should be presented that could be quickly grasped, and that a more detailed overview would be useful later on.

Presented simple and funny visualizations included smiley's or a desktop picture with flowers, a sun and clouds which depict how well your state of working is. Moreover an interactive character was presented, that greets you when you start your work, reminds you to start working on a project when you are wasting time in your inbox, gives you relevant news facts from time to time and asks you whether you wish to see your stats or share your mood state via Twitter. The user can communicate his mood to the character by petting or hitting it.

More technical visualizations with facts in detail include various diagrams depicting what you have done and your state. It was noted that the user himself should be able to select which things to be visualized in the graphs. Between participants a difference was seen whether the pure data should be visualized (e.g. amount of application switches) and interpreted by the user himself or whether the software should interpret low level data and present aggregated scores (e.g. measure of focus). All in all it turned out that the specific visualization preferences are very different among people and that personalization is a very important aspect.

In general many people preferred a playful manner of the tool. Some games to help you improve your behaviour were presented. For example a flow meter that is depicted in the task bar, which gets fuller the less you switch tasks, which motivates to work focused. When the meter is full a mini game may be played as reward.

Session 3 – When and how is the self management tool used?

In the interviews the people that had experienced burnout where asked how and when they would use the self management tool. In general people wished to personally adjust which aspects to monitor and visualize and also wished to adjust how often the tool gives feedback. Moreover they wanted to be able to turn the tool on and off themselves and be able to delete data. The logged data should not be shared with the management but used for individual purposes only. Finally the tool should work on all work locations or computers of the user.

Because of time constraints this session was not explicitly carried out in the workshop. Nevertheless during other sessions some convenient moments at which the system could give feedback were mentioned: when you start your day, after the break or when you finish your day. Moreover it was noted that the tool could become active and warn you or intervene when it detects problematic states.

Session 4 – Which aspects have to be taken in mind when introducing the tool?

In the last session we wanted the participants to come up with reasons for using or not using the system. Again first the results of the interviews were presented and participants were asked to add further reasons.

The interviewed people explained that they would use the system when the ease of installation and the ease of use is high and when they have a feeling of having control over the tool. Moreover they want to have a clear picture of what the tool does and how their data is used. Privacy concerns and the possibility to delete data play a great role.

The workshop participants further mention that they would only use the tool when it yields benefits, for example improving their well being or working style or helping them filling in SAP time sheets. It should be clearly communicated that the tool is not going to irritate the user with pop ups and can completely be personalized to his needs. One participant notes that the tool should only present facts and not draw conclusions for him. What most people would convince to use the tool is a funny and playful manner. A game element could motivate people to actually change their behavior.

Reasons for not using the tool given by the workshop participants were the following: when they have the feeling of not needing the tool or when the tool runs so in the background that they simply forget it. People will also not use the tool when it costs too much time, when other people can track their data and when they have the feeling the system gives wrong information. Finally, users do not like it when the system is too confronting or when it intervenes too much.

Conclusion

All in all the workshop together with the interviews yielded many insights and a variance of interesting ideas. The most important conclusions are that personalization plays an important role and that a funny and playful style could convince people to use the self management tool. In general it is important that users get a clear picture of the possibilities and benefits of the software. Concerns about privacy and loss of control are the most important aspects to address when introducing the tool.