Detecting Memory-Based Interaction Obstacles with a Recurrent Neural Model of User Behavior

Felix Putze*
University of Bremen
Bremen, Germany
felix.putze@uni-bremen.de

Mazen Salous
University of Bremen
Bremen, Germany
mazen.salous@uni-bremen.de

Tanja Schultz
University of Bremen
Bremen, Germany
tanja.schultz@uni-bremen.de

ABSTRACT
A memory-based interaction obstacle is a condition which impedes human memory during Human-Computer Interaction, for example a memory-loading secondary task. In this paper, we present an approach to detect the presence of such memory-based interaction obstacles from logged user behavior during system use. For this purpose, we use a recurrent neural network which models the resulting temporal sequences. To acquire a sufficient number of training episodes, we employ a cognitive user simulation. We evaluate the approach with data from a user test and on which we outperform a non-sequential baseline by up to 42% relative.

CCS Concepts
•**Human-centered computing** → **User models;**

Author Keywords
Classification of user behavior; memory; interaction obstacles; LSTMs

INTRODUCTION
When humans interact with computers, they may face interaction obstacles which constrain their ability to successfully complete their tasks. Interaction obstacles occur in various forms, related to perceptual, cognitive, or motor challenges. For an adaptive interaction system, modeling such interaction obstacles is important to provide individual support targeting these interaction obstacles. One of the central human cognitive functions is working memory. Working memory is strongly correlated to general intelligence [4] and determines performance in most non-trivial HCI tasks. If working memory of the user is impeded, an intelligent system could for example adapt by reducing its own memory load [25].

Existing approaches (see Section 2) for dynamically detecting memory-based interaction obstacles online make use of one or more sensors to classify the cognitive state of the user. In many situations however, using sensors may not be feasible due to cost, convenience, or privacy considerations. In this paper, we present an approach for detecting a memory-loading secondary task as an interaction obstacle purely by exploiting behavioral patterns from interaction features without any additional sensors. To this end, we employ Long Short Term Memory (LSTM) networks, which are a special type of recurrent neural networks to model temporal relationships in the sequence of behavioral patterns.

For analyzing this approach, we investigate data recorded during the play of a game of matching pairs. Matching pairs is a stand-in for a complex interaction task with a strong memory component. Matching pairs is also a relevant study subject in its own right as it is used for the activation of people suffering from dementia [6], which is a condition that strongly affects memory. In this work, our goal is to identify the presence of memory-based interaction obstacles in the form of a memory-loading secondary task by classifying sequences of user behavior data, recorded as interaction features. We expect players to behave differently based on whether their memory is distracted by a secondary task, e.g., by missing opportunities to collect previously revealed pairs of cards or revealing the same card multiple times. We will show that, by using a sequential classification model of human behavioral data, we are able to discriminate between game play situations with and without secondary task.

Our two main contributions in this paper are: First, we present a novel LSTM-based approach for the detection of memory-based interaction obstacles from sequential behavioral data. Second, for dealing with data sparseness in modeling user behavior from real data, we present a novel approach for the cognitively motivated simulation of user behavior to generate enough training sequences.

RELATED WORK
As working memory is such an important cognitive construct influencing many HCI tasks, there exists a large body of research which aims at measuring working memory load [21], which is usually modulated through memory-loading secondary tasks. Actual measurement of working memory load then takes place by classifying features from various sensors. Most prominently, researchers investigate signals from neural sources, for example recorded from electroencephalography (EEG): Berka et al. [3] showed the feasibility of classifying working memory load from frequency features at different electrode sides using an unobtrusive EEG headset. Baldwin

*Partially funded by DFG grant PU 613/1-1: Detection of Interaction Competencies and Obstacles (DINCO).

IUI'18, March 7–11, 2018, Tokyo, Japan

ISBN 978-1-4503-4945-1/18/03…$15.00
DOI: 10.1145/3172944.3173006

and Penaranda [2] used a classifier based on Neural Networks to discriminate two difficulty levels across three different memory tasks. Ke et al. [16] showed that a memory load classifier based on a broad range of frequency features could be transferred from a simple n-back task to the more complex Multi-Attribute Task battery. Herff et al. [13] showed how a hybrid classifier (combining EEG and fNIRS measurement) was able to reliably discriminate pairs of difficulty levels in a sequential memory task. Mühl et al. [18] used Common Spatial Patterns on different frequency ranges to generate features for classification of workload under different stress conditions. Heger et al. [12] showed that adapting the behavior of a system to workload measured from EEG in a multitasking situation helped users to improve efficiency and effectiveness in the main task. Brain activity is not the only modality which carries information on memory load; eye tracking data can also be used for this purpose: Rozado and Dünser [22] demonstrated that EEG-based load estimation can be combined with pupillometry data for increased accuracy. Katidioti et al. [15] used pupillometry-based load estimation to create a task-independent interruption management system.

All of the presented approaches have in common that they only regard relatively short segments of sensor data without resorting to a larger temporal context. Recurrent neural networks as sequential classification models allow to take such information into account. An LSTM network is a type of recurrent neural network which uses designated memory cells to model long-range dependencies within sequential data. For example, Gers et al. [8] showed that LSTMs were able to successfully learn the rules encoded in context-free and context-dependent formal languages. Schaul and Schmidhuber [23] investigated the use of LSTMs for modeling playing strategies in complex games and the ability of the model to generalize from small to larger game boards. Graves et al. [10] used bidirectional LSTMs for recognition of unconstrained, connected handwriting. Eck and Schmidhuber [5] showed that LSTMs were able to discover and reproduce the temporal structure of complex pieces of music due to their ability in detecting long-range dependencies. While LSTM has been established as a powerful modeling tool, it has to our best knowledge never been applied to behavioral data in the HCI context.

DATA COLLECTION

The well-known matching pairs game, constitutes a complex HCI task which relies on working memory. In this game, a player in one turn reveals two cards from a display of face-down cards. If the revealed cards match (i.e., show the same picture), they are removed from the game, otherwise, they are turned face-down again. The goal of the game is to clear the display with as few turns as possible, by memorizing the location of already revealed cards. How people approach this task carries a lot of information about their working memory status. The game used in our data collection experiments consists of randomly shuffled fourteen cards (seven pairs) and was presented on an Android tablet, see Figure 1. Participants played a single-player variant of the game in which they had the goal of clearing the display as fast as possible. The game application records log files of interaction features, such as position and identity of selected cards.

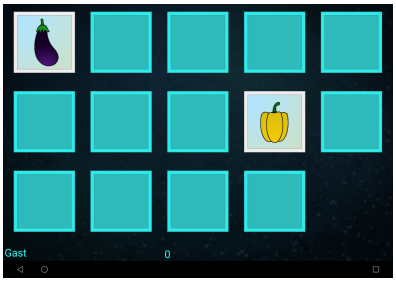

Figure 1. Matching Pairs Game

To create a memory-related interaction obstacle, we introduced a secondary task to the matching pairs game. Participants were asked to play the game two times with and without secondary task, in randomized order:

- Only Matching Pairs (MP): The participants are asked to play the game without any parallel secondary task.
- Matching Pairs with Cumulative Sum (MP+CS): Whenever the participant reveals a card, a random number between 1 and 9 will be spoken by the synthesized voice. The participant is asked to calculate and memorize the sum of all spoken numbers throughout the game.

In total, 31 people participated in our experiments. 22 participants were male and the other 9 participants were female. The ages range from 19 till 48, most of the participants were students or university employees. All participants gave their informed written consent. They were compensated with 10€ for their participation. The data collection was approved by the ethics committee of the University of Bremen. For a subset of 24 participants, we conducted the Memory Update Test (MU) [17] in order to enable further evaluation of our paradigm in comparison to a standard working memory test. For the same subset of participants, we also recorded EEG data, but this data is not evaluated in the current paper.

CLASSIFICATION SETUP

In this section, we will introduce the classification setup which we implemented for the detection of memory-based interaction obstacles. We treat the recorded data as a binary classification task, where the output label is either MP or MP+CS. For this purpose, we compare two classification models: A sequential neural model based on LSTMs [14] and a static baseline model based on Linear Discriminant Analysis (LDA). As the number of available training episodes is very limited (each session yields two episodes, one of each class), we generate additional training data from a cognitive user simulation. We train the classifier from data of 15 participants. For each participant, we generate 1,000 episodes, resulting in a total of 15,000 training episodes.

The overall system consists of two main components: The Cognitive Memory Model (CMM) [19], which is used to simulate plausible additional training episodes, and the classifier which is trained to label given game sequences (or prefixes thereof) as either MP or MP+CS. The following subsections explain the role of both parts.

CMM-based User Simulation

A challenge of generating additional training sequences for training of a sequential classification model is that the sequences have to maintain plausible temporal relationships between time slices. While traditional oversampling approaches, such as ADASYN [11], do not consider the temporal structure of sequences, recent approaches, such as [9], aim at capturing this temporal structure to generate plausible sequential data.

The Cognitive Memory Model [19] (CMM) is a general computational cognitive model of human memory inspired by the ACT-R theory [1]. It has been successfully employed to model games of matching pairs, revealing different playing strategies and levels of memory performance. In this work, the CMM acts as a simulator to generate additional training episodes for the classifier to detect memory-based interaction obstacles. The CMM is based on a decay-based forgetting mechanism to realistically simulate non-perfect human memory. For each item in memory (cards in our use case), the CMM maintains an activation value from which the likelihood of retrieval can be calculated. In general, the activation of an item depends on the frequency and recency of stimulations of said item. Based on the retrieval likelihood for all individual cards in a given game state, we execute a specific strategy to select which cards to reveal. This strategy balances exploration (revealing cards which are unknown to the player or not remembered) and exploitation (revealing pairs when position of both corresponding cards is remembered with high enough probability). By repeatedly executing this strategy and updating the game state, artificial sequences of game play can be generated. See the work by Putze et al. [20] for details on how to model the game of matching pairs using the CMM.

The CMM has a number of free parameters, such as the degree of memory decay, which determine its predictions of memory performance. Those parameters can be optimized by a genetic optimization algorithm in CMM to best fit the empirical training data. The genetic optimizer maintains a population of CMM configurations consisting of all free parameters. The population is initialized randomly and then iteratively updated according to the standard operations of genetic optimization, mutation and selection [7]. For selection, we employ two different similarity metrics, the so-called matching-pairs and exploiting-punishment measures, to compare a set of generated game sessions to real game sessions. The matching-pairs measure counts the number of removed card pairs for each turn. The exploiting-punishment measure counts the number of card pairs which have both been revealed at some point of the game but have not been removed by the player (presumably, because the location of one or both of the cards to that pair have been forgotten by the player). A CMM configuration is then selected more likely if it generates game sessions which are close to the real data according to the specified similarity metrics. As we are comparing sequences, the similarity measures are also defined per game turn and distance between two game sessions is measured as the Root Mean Squared Error (RMSE) between the two turn-wise sequences.

Playing behavior in the game of matching pairs varies between individual players, for example depending on working memory capacity and differences in playing strategy. To achieve enough variance in the training data, we do not pool all available real sessions together to optimize a global parameter set; instead, we repeat the process for each session individually and only pool the resulting simulated sessions together to form a richer, more varied corpus of simulated sessions.

Classifiers

LSTMs are a special form of recurrent neural network which are designed to model sequential data. LSTMs consist of so-called LSTM cells which explicitly store, retain, or forget information from previous time steps. This approach is chosen to battle the challenge of vanishing gradient in traditional recurrent neural networks. The network for our approach is shown in Figure 2. It starts with an LSTM layer with 32 LSTM

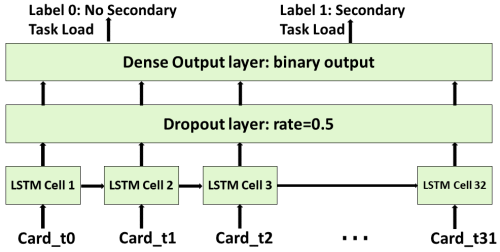

Figure 2. Prediction model bottom-up topology

cells as input layer to handle the training data sequences. The LSTM is followed by a Dropout layer with rate of 0.5 acting as a regularization technique that avoids over-fitting by randomly ignoring neurons during training [24]. The output layer is a fully connected dense layer that uses soft-max activation and outputs a binary label. To fit the model, we perform 500 epochs of Stochastic Gradient Decent (SGD) which has been used with adaptive learning rate $lr = \frac{0.1}{\#epoch}$. The input sequences consist of the ordered consecutive chosen cards, each card represented by two features: The first one encodes its position in the revealing order of motives (from 1 to 7, since the game has 14 cards, i.e., 7 pairs), while the second feature encodes the card's position in the corresponding pair (1 or 2). After 500 training epochs on simulated data, we retrain the resulting model with the original 15 real training sessions.

In contrast to the LSTM model, the LDA model considers a whole game at once. For this purpose, we need to manually define the features which are expected to differ between classes. The following features summarize for a game prefix how efficient and close-to-optimal a player performed: 1) Number of cards left in the game, 2) Number of never revealed cards in the game, 3) Maximum number of times revealing the same card, and 4) Number of turns since game completion ($= 0$ if game not yet completed). From these features, we train an LDA model to classify logged games into MP and MP+CS.

EVALUATION

Before analyzing the trained classifiers, we check the behavioral differences between classes MP and MP+CS. Figure 3 shows the matching-pairs statistic for both classes. We see a difference between the two conditions, although standard deviation is high. This implies that the secondary task deteriorates MP performance but its presence cannot immediately detected from simple behavioral measurements[1]. We also compared performance in MP for participants with above-median and below-median memory capacity according to the MU test and saw a significant difference in game completion time. This confirms that the MP task is actually memory-dependant.

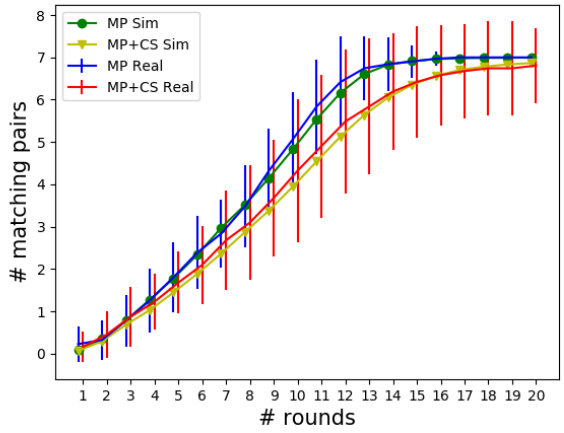

Figure 3. Mean matching-pairs statistic for MP and MP+CS conditions of real data (*MP Real* and *MP+CS Real*) and simulated data (*MP Sim* and *MP+CS Sim*). For real data, whiskers show standard deviation.

As a next step, we compare the simulated sessions to the real ones. Figure 3 also presents the mean matching-pairs metric of all generated sessions. The evident similarity can also be quantified by calculating turn-wise RMSE between curves for corresponding real and simulated curves. Simulated data from class MP yields an RMSE of 0.12 to the real data from class MP and an RMSE of 0.41 from class MP+CS. Distance is similarly large for simulated data from class MP+CS (RMSE of 0.67 to real MP and 0.21 to real MP+CS).

For evaluating the classifiers, we repeatedly split the available 31 playing sessions randomly into training and testing sets, of size 15 and 16, respectively. For each split, we again performed multiple "inner" iterations with different randomly generated sets of 15000 training sessions. In this setting, we used 20 splits with 20 inner iterations each. Training and testing data is balanced for classes MP and MP+CS. We test the trained classification model on the testing data of the respective split, as well as on 3000 newly generated sessions which are disjoint from the training sessions. We perform the analysis for game prefixes of different fixed lengths. Since our matching pairs game has 7 pairs of cards, the minimum game length is 7 turns. In addition, all participants' logs show that the participants finish the game in 10 to 15 game turns.

[1]the matching-pairs statistic corresponds to the first LDA feature.

Length	*LSTM Sim*	*LDA Sim*	*LSTM Real*	*LDA Real*
7	58.2 (0.9)	52.6 (0.9)	62.6 (1.1)	43.9 (1.3)
10	59.3 (0.8)	50.0 (0.1)	66.1 (0.5)	56.2 (0.0)
15	60.1 (0.5)	56.8 (0.8)	65.1 (1.3)	57.9 (2.3)

Table 1. Average accuracy (and standard error) for LSTM and and LDA classification models on simulated and real testing data for game prefixes of different lengths.

Therefore, we look at game prefixes of length 7, 10, and 15. As performance metric, we report classification accuracy, summarized in Table 1.

These results show that the LSTM model outperforms the LDA model. The improvement is statistically significant for all results on simulated and real data ($p < 0.05$, calculated using a paired t-test on the result of individual iterations). The LSTM model outperforms the LDA baseline by more than 42% relative for 7 game turns, 17% relative for 10 game turnes and more than 12% relative for 15 game turns. Such results show high advantages of using LSTM over LDA especially for short behavioural sequences (7 game turns). While the manual defined LDA features require relatively long game prefixes to estimate reliable game statistics, the LSTM exploits temporal dependencies also from shorter game prefixes. However, the model is still in-progress and these results can be further improved by tuning the hyper-parameters and generating more training data.

DISCUSSION

In this paper, we have demonstrated the feasibility of detecting memory-based interaction obstacles without resorting to additional sensors by modeling time series of user behavior with sequential LSTMs. For generating additional training episodes, we employed a novel, cognitively inspired simulation and showed their similarity to the real data. We showed that the simulation was able to generate plausible training sessions and that the sequential LSTM-based model outperformed a static LDA-based baseline by up to 42% relative. Although classification needs to be further improved to be reliable enough for adaptive HCI systems, the approach is promising to be helpful for a large variety of scenarios: Cognitive architectures such as ACT-R, which is the main source for the employed simulation, aim at modeling general human behavior. Therefore, we assume that the approach for classifying sequences of human behavior data can be transferred to other tasks by adjusting the simulation. For this purpose, it is also relevant that it is not necessary to engineer features for the LSTM which works with the raw sequence of user behavior. Next steps will focus on the exploitation of the additional data recorded in this study: 1) We will investigate the recorded EEG data for the detection of memory-based interaction obstacles from combined neural and behavioral features. 2) We will use the data from the MU test to investigate memory-based interaction obstacles which result from limited working memory capacity (in contrast to memory-loading secondary tasks).

REFERENCES

1. John R. Anderson, Daniel Bothell, Michael D. Byrne, Scott Douglass, Christian Lebiere, and Yulin Qin. 2004.

An Integrated Theory of the Mind. *Psychological Review* 111, 4 (2004), 1036–1060.

2. Carryl L Baldwin and BN Penaranda. 2012. Adaptive training using an artificial neural network and EEG metrics for within-and cross-task workload classification. *NeuroImage* 59, 1 (2012), 48–56.
3. Chris Berka, Daniel J. Levendowski, Michelle N. Lumicao, Alan Yau, Gene Davis, Vladimir T. Zivkovic, Richard E. Olmstead, Patrice D. Tremoulet, and Patrick L. Craven. 2007. EEG Correlates of Task Engagement and Mental Workload in Vigilance, Learning, and Memory Tasks. *Aviation, Space, and Environmental Medicine* 78 (2007), B231–B244.
4. Andrew RA Conway, Michael J Kane, and Randall W Engle. 2003. Working memory capacity and its relation to general intelligence. *Trends in cognitive sciences* 7, 12 (2003), 547–552.
5. Douglas Eck and Juergen Schmidhuber. 2002. Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. In *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on*. IEEE, 747–756.
6. S Ehret, F Putze, H Miller-Teynor, A Kruse, and T Schultz. 2017. Technique-based game for daycare visitors with and without dementia: Effects, heuristics and correlates. *Zeitschrift fur Gerontologie und Geriatrie* 50, 1 (2017), 35–44.
7. David B Fogel. 1994. An introduction to simulated evolutionary optimization. *IEEE transactions on neural networks* 5, 1 (1994), 3–14.
8. Felix A Gers, Jrgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with LSTM. *Neural computation* 12, 10 (2000), 2451–2471.
9. Zhichen Gong and Huanhuan Chen. 2016. Model-Based Oversampling for Imbalanced Sequence Classification. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 1009–1018.
10. Alex Graves, Marcus Liwicki, Santiago Fernndez, Roman Bertolami, Horst Bunke, and Jrgen Schmidhuber. 2009. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence* 31, 5 (2009), 855–868.
11. Haibo He, Yang Bai, E. A. Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. 1322–1328.
12. Dominic Heger, Felix Putze, and Tanja Schultz. 2010. An adaptive information system for an empathic robot using EEG data. In *Social Robotics*. Springer, 151–160.
13. C. Herff, O. Fortmann, Chun-Yu Tse, Xiaoqin Cheng, F. Putze, D. Heger, and T. Schultz. 2015. Hybrid fNIRS-EEG based discrimination of 5 levels of memory load. In *2015 7th International IEEE/EMBS Conference on Neural Engineering (NER)*. 5–8.
14. Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
15. Ioanna Katidioti, Jelmer P Borst, Douwe J Bierens de Haan, Tamara Pepping, Marieke K van Vugt, and Niels A Taatgen. 2016. Interrupted by Your Pupil: An Interruption Management System Based on Pupil Dilation. *International Journal of Human–Computer Interaction* 32, 10 (2016), 791–801.
16. Yufeng Ke, Hongzhi Qi, Feng He, Shuang Liu, Xin Zhao, Peng Zhou, Lixin Zhang, and Dong Ming. 2014. An EEG-based mental workload estimator trained on working memory task can work well under simulated multi-attribute task. *Frontiers in human neuroscience* 8 (2014).
17. Stephan Lewandowsky, Klaus Oberauer, Lee-Xieng Yang, and Ullrich KH Ecker. 2010. A working memory test battery for MATLAB. *Behavior Research Methods* 42, 2 (2010), 571–585.
18. Christian Mühl, Camille Jeunet, and Fabien Lotte. 2014. EEG-based workload estimation across affective contexts. *Frontiers in neuroscience* 8 (2014).
19. Robert Pröpper, Felix Putze, and Tanja Schultz. 2011. JAM: Java-based Associative Memory. In *Proceedings of the Paralinguistic Information and its Integration in Spoken Dialogue Systems Workshop*. Springer, 143–155.
20. Felix Putze, Sonja Ehret, Heike Miller-Teynor, Andreas Kruse, and Tanja Schultz. 2015. Model-based Evaluation of Playing Strategies in a Memo Game for Elderly Users. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*. Hongkong, China.
21. Felix Putze and Tanja Schultz. 2014. Adaptive cognitive technical systems. *Journal of neuroscience methods* 234 (2014), 108–115.
22. David Rozado, Andreas Duenser, and Ben Howell. 2015. Improving the performance of an EEG-based motor imagery brain computer interface using task evoked changes in pupil diameter. *PloS one* 10, 3 (2015), e0121262.
23. Tom Schaul and Jürgen Schmidhuber. 2009. Scalable Neural Networks for Board Games. In *Artificial Neural Networks ICANN 2009*. 1005–1014.
24. Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15 (2014), 1929–1958.
25. Maria Wolters, Kallirroi Georgila, Johanna D Moore, Robert H Logie, Sarah E MacPherson, and Matthew Watson. 2009. Reducing working memory load in spoken dialogue systems. *Interacting with Computers* 21, 4 (2009), 276–287.