Dealing with Concept Drift in Exploratory Search: An Interactive Bayesian Approach

Antti Kangasrääsiö
Helsinki Institute for Information Technology HIIT
Department of Computer Science
Aalto University
antti.kangasraasio@hiit.fi

Yi Chen
Helsinki Institute for Information Technology HIIT
Department of Computer Science
Aalto University
yi.chen@hiit.fi

Dorota Głowacka
Helsinki Institute for Information Technology HIIT
Department of Computer Science
University of Helsinki
dorota.glowacka@hiit.fi

Samuel Kaski
Helsinki Institute for Information Technology HIIT
Department of Computer Science
Aalto University
samuel.kaski@hiit.fi

IUI'16 Companion, March 07–10, 2016, Sonoma, CA, USA
ACM 978-1-4503-4140-0/16/03.
http://dx.doi.org/10.1145/2876456.2879487

Abstract
In exploratory search, when the user formulates a query iteratively through relevance feedback, it is likely that the feedback given earlier requires adjustment later on. The main reason for this is that the user learns while searching, which causes changes in the relevance of items and features as estimated by the user – a phenomenon known as *concept drift*. It might be helpful for the user to see the recent history of her feedback and get suggestions from the system about the accuracy of that feedback. In this paper we present a timeline interface that visualizes the feedback history, and a Bayesian regression model that can estimate jointly the user's current interests and the accuracy of each user feedback. We demonstrate that the user model can improve retrieval performance over a baseline model that does not estimate accuracy of user feedback. Furthermore, we show that the new interface provides usability improvements, which leads to the users interacting more with it.

Author Keywords
Concept drift; Exploratory search; Interactive User Modeling; Probabilistic User Models; User interfaces

ACM Classification Keywords
H.3.3 [Information Search and Retrieval]: Relevance feedback; H.5.2 [Information Interfaces and Presentation]: User Interfaces

Introduction

In exploratory search, the user initially has some knowledge of the search topic but not enough knowledge to reduce the task into a simple fact retrieval [6]. Thus, the user has to learn while searching, iteratively reformulating a hypothesis of what information would satisfy her information need and where to find it. Naturally, this type of setting makes it difficult for the user to directly formulate good search queries.

A recently developed search system called *SciNet* [3, 4] aims to solve this issue by allowing the user to interactively formulate her search query, starting from a general keyword query, and improving it by interactive relevance feedback. However, the system does not take concept drift [2, 9] into account[1]. In this paper, we improve over the existing system by formulating a user model that can deal with concept drift and we develop an interface that allows the user to interact with this new model.

The user model is a Bayesian regression model that estimates both the current search intent of the user and the accuracy of the relevance feedback provided by the user. For collecting feedback we introduce a timeline interface that visualizes the user's recent feedback history[2]. The interface highlights the past feedbacks that were estimated inaccurate and allows the user to interact with the visualized keywords. The introduced model is able to improve retrieval accuracy in a simulation experiment. In a user study, users interact more with the new interface[3] and report usability improvements in interviews.

The User Model

We assume that the user's interest can be approximately described with a linear Gaussian model, where the accuracy of feedback given by the user may be different for each observation. This gives us the model $y_i \sim N(x_i\phi, \sigma^2/w_i)$, where y_i is the user-given relevance score to an item with features x_i; ϕ is the linear interest model with variance σ^2, and w_i is the accuracy of observation i. We assume prior distributions: $\phi_j \sim N(\mu_\phi, \lambda_\phi)$, $\sigma^2 \sim \Gamma^{-1}(\alpha_{\sigma^2}, \beta_{\sigma^2})$, $w_i \sim \Gamma(\alpha_{w_i}, \beta_{w_i})$, $w_i^{fix} \sim \delta(1.0)$, where ϕ_j is the jth component of the vector ϕ. We also allow the user to correct the inferences by forcing certain feedback to be treated as accurate; in this case we use w_i^{fix} instead of w_i, making the accuracy for that feedback be always 1.0.

We will refer to this model as the *ARD model*, as the determination of observation weights can be seen as Automatic Relevance Determination [5]. The posterior of the parameters is estimated using mean-field variational inference [1]. A similar model has been used successfully for outlier detection in robotics [8][4].

User Interface

We developed a timeline interface to visualize the user's history of relevance feedback (Figure 1). The most recent feedback appears at the top of the timeline. On the right hand side of the timeline, the relevance of each keyword is visualized by both length and color of the bar[5]. Keywords estimated to require revision are made more salient by a red question mark icon[6]. The user can adjust the relevance value of a feedback by dragging the bar. She can also indicate that a feedback is accurate by clicking the lock icon or remove a feedback by clicking the X-icon.

In longer sessions, in particular when feedback is given iteratively, it is likely that the user will not remember details of all the previous feedback. The timeline allows the user to evaluate the feedback given so far and make changes to it if needed. Highlighting the feedback likely in need of revision is assumed to help the user to find feedback in need of revision more easily. The option to react to both true and

[1]The user model in this system makes the assumption that all user feedback is equally accurate, the user makes no mistakes in giving feedback and that no learning or change in search interests would occur.

[2]User indicates relevancy of keywords by giving feedback within the range $[0.0, 1.0]$, where 1.0 is highly relevant.

[3]More feedback improves the quality of the user model.

[4]Our model allows the user to correct the inferences, and it estimates σ^2 with variational inference instead of using a point estimate. Taking full distributions into account is important as only a small amount of data is available for fitting the model.

[5]Relevant keywords have long green bars, whereas irrelevant ones have short red bars.

[6]Feedback is highlighted when the estimated accuracy w_i is below a threshold value of 0.65. The value was hand-tuned.

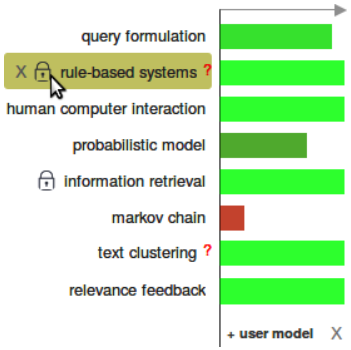

Figure 1: The timeline interface visualizes past feedback and provides the user with ways to interact with it. Feedback most likely in need of revision is highlighted with red question mark icons.

[7]L2-normalized TF-IDF feature vectors of length 539 were generated for the posts (terms with document frequency between 0.2 and 0.04 were used).

[8]After 10 steps ARD had average runtime of 0.6 s, whereas LG had 0.4 s (wall clock time). After 100 steps the average runtimes were 1.4 s for ARD and 0.8 s for LG.

[9]$\mu_\phi = 0.0$, $\lambda_\phi = 0.1$, $\alpha_{\sigma^2} = 2.5$, $\beta_{\sigma^2} = 0.5$, $\alpha_w = 0.7$, $\beta_w = 1.0$.

false highlights (by adjusting keyword relevance and marking feedback as accurate) was motivated by the results of the simulation experiment. Keywords from previous search sessions are added as expandable lists at the bottom of the timeline. This feature provided the user with a way to re-find keywords that the user has given feedback to previously.

The interface is otherwise similar to that presented in [4]. There is a radar interface on the left side of the screen, displaying the current state of the search intent model to the user. The user can adjust the model by moving keywords on the radar. The timeline is situated under the radar and the list of most relevant results is to the right of the radar.

Simulation Experiment

To study the performance of the user model, we conducted an experiment with a simulated user. As a dataset we used the 20 Newsgroups dataset [7], containing 2000 posted messages, 100 from each of 20 newsgroups[7].

In each repeated experiment the simulated user selected at random one of the newsgroups and initialized the search by indicating two positive messages at random. The user then saw a list of 50 most relevant documents and, in some scenarios, one highlighted past feedback the user should re-evaluate. The user then replied by giving noisy feedback to one item in the list and by revising the highlighted feedback. The F1-score of the list of 50 items was recorded at every step (representing the quality of found items)[8].

The noisy feedback was generated as follows. 70% of the time a positive example was selected and relevance feedback 1.0 was given to it; 10% of the time a negative example and feedback 0.0. 20% of the time the user would select a random item from the list and give it relevance feedback 1.0 with 87.5% probability and 0.0 with 12.5% probability.

The experiment was repeated in four different scenarios. In Scenario A, no items were highlighted to the user and the user made no revisions to the previous feedback. In Scenario B, the user revised the highlighted feedback if it did not have the correct relevance value (revising true positives) and marked it as accurate if it already had the correct relevance value (indicating false positives). In Scenario C, the user only revised true positives, and in Scenario D the user only indicated false positives.

We compared the performance of the ARD model to a baseline and an oracle. The baseline is a Linear Gaussian model that is otherwise similar to the ARD model, except that all feedback is equally accurate ($w_i = 1.0$). We will call this the *LG model*. The *Oracle* only uses the relevant observations for fitting the model, being otherwise similar to the LG model. Model parameters were hand-tuned over a small number of iterations to avoid over-fitting[9].

The highlighted items were chosen by selecting the feedback with the lowest w_i value. Draws were resolved randomly. LG model sampled items to highlight uniformly and Oracle highlighted based on the true relevance values.

The retrieval performance is shown in Figure 2. We observed that the ARD model performs similarly to the LG model if no corrections are made to the historical feedback. If the user made corrections to all the highlighted feedback, the performance of the ARD model approaches that of the Oracle. If the user made corrections to only the true positive highlighted items, the improvements were still noticeable but smaller. If the user made corrections to only the false positive highlighted items, the improvements were small.

User study

We ran a user study where we compared the new interface with a baseline where the timeline was hidden. The ARD

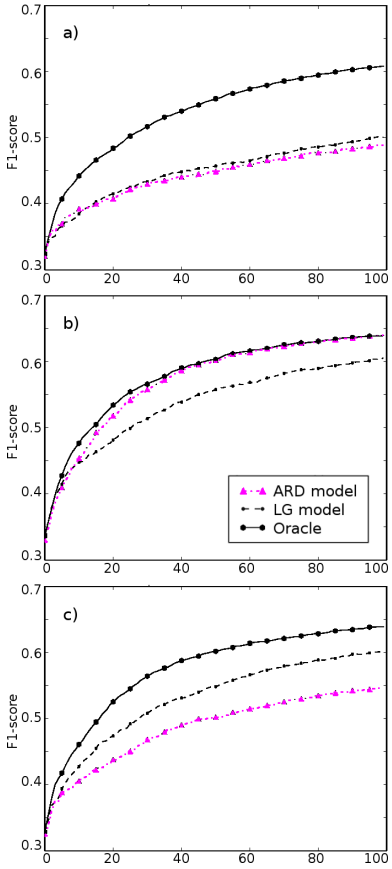

Figure 2: F1-scores in 100 iterations, averaged over 200 experiments. **a)** Scenario A, **b)** Scenario B, **c)** Scenario C. Scenario D was similar to a).

[10] $\mu_\phi = 0.0$, $\lambda_\phi = 0.1$, $\alpha_{\sigma^2} = 2.0$, $\beta_{\sigma^2} = 0.1$, $\alpha_w = 1.0$, $\beta_w = 1.0$.

model was used in both cases as based on the simulation experiment the performance should be similar to the LG model when there are no timeline interactions. Parameters were tuned by hand over a small number of iterations[10].

Each participant performed two sets of tasks – two tasks in each set, one with each interface. The order of the sets and tasks within sets was balanced as was the matching of interfaces to tasks. The duration of each task was 20 minutes. In the first set of tasks the user was asked to write a short draft of an essay on a given topic. In the second set of tasks, the user was asked to conduct free search on their own research topic. The study had four participants: two first-year PhD students and two MSc students.

The users performed on average four keyword queries per task with both interfaces. However, the number of keyword-related interactions (giving feedback to a keyword, removing or marking feedback as accurate) was larger with the new interface. Users did on average 5.4 keyword interactions per task with the baseline and 8.9 with the new interface ($p = 0.22$). The interactions with the new interface consisted of on average 5.6 keyword feedback on the radar, 1.0 keyword feedback on the timeline, 1.6 keyword deletions from the timeline and 0.6 feedback marked as accurate.

After each task set, we conducted a semi-structured interview with the user. The following main benefits of the timeline interface were reported: i) It is easier to understand what feedback affects the results as it is visualized on the timeline; ii) It is easy to re-find keywords both from current and past search sessions; iii) It helps in the search process as it is easy to "go back" by deleting suitable keywords.

The users also reported the following drawbacks: iv) The red question mark icon made the user feel as if she had made an error; v) The user felt like the system was not fully under her control when the accuracy of feedback was changed automatically; vi) The task time limit made them avoid functionality they were not familiar with, as their focus was on performing the task well.

Conclusion

We have presented a user model that is able to take into account concept drift when a search intent model is refined iteratively. In a simulation study, the performance of the model was better than a baseline that does not model the variation in accuracy of the user feedback. When the user reacted to both true and false recommendations on what feedback to adjust, the model approached the performance of an oracle. Even if the user only reacted to part of the suggestions, the performance was better than the initial baseline.

We have also presented a timeline interface that offers the user suggestions on what past feedback is most likely in need of adjustment. In a user study, we found indication that the new interface elicited more user interactions with the system. The users reported that the timeline made it easier to understand what feedback affects the search results, helps to re-find keywords, and provides help in returning back to a previous state in the search process.

To the best of our knowledge, this is the first time a system has been presented that both models the accuracy of individual user feedback in a search setting and allows the user to directly interact with this model.

Acknowledgments

This work was supported by the Academy of Finland grant 251170 (COIN), TEKES (Re:Know), EU Seventh Framework Programme (FP7/2007-2013) grant 611570 and computational resources provided by the Aalto Science-IT project.

REFERENCES

1. Hagai Attias. 1999. Inferring Parameters and Structure of Latent Variable Models by Variational Bayes. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99)*. Morgan Kaufmann Publishers Inc., 21–30. http://dl.acm.org/citation.cfm?id=2073796.2073799
2. João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A Survey on Concept Drift Adaptation. *Comput. Surveys* 46, 4, Article 44 (2014), 37 pages. DOI: http://dx.doi.org/10.1145/2523813
3. Dorota Glowacka, Tuukka Ruotsalo, Ksenia Konuyshkova, Kumaripaba Athukorala, Samuel Kaski, and Giulio Jacucci. 2013. Directing Exploratory Search: Reinforcement Learning from User Interactions with Keywords. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces (IUI '13)*. ACM, 117–128. DOI: http://dx.doi.org/10.1145/2449396.2449413
4. Antti Kangasrääsiö, Dorota Glowacka, and Samuel Kaski. 2015. Improving Controllability and Predictability of Interactive Recommendation Interfaces for Exploratory Search. In *Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI '15)*. ACM, 247–251. DOI: http://dx.doi.org/10.1145/2678025.2701371
5. David J. C. MacKay. 1994. Bayesian Nonlinear Modeling for the Prediction Competition. *ASHRAE transactions* 100, 2 (1994), 1053–1062.
6. Gary Marchionini. 2006. Exploratory Search: From Finding to Understanding. *Commun. ACM* 49, 4 (2006), 41–46. DOI: http://dx.doi.org/10.1145/1121949.1121979
7. Jason Rennie and Ken Lang. 2008. The 20 Newsgroups Dataset. (14 January 2008). http://qwone.com/~jason/20Newsgroups/.
8. Jo-Anne Ting, Aaron D'Souza, and Stefan Schaal. 2007. Automatic Outlier Detection: A Bayesian Approach. In *In IEEE International Conference on Robotics and Automation*. 2489–2494. DOI: http://dx.doi.org/10.1109/ROBOT.2007.363693
9. Alexey Tsymbal. 2004. *The Problem of Concept Drift: Definitions and Related Work*. Technical Report. Computer Science Department, Trinity College Dublin.