A Federated Search and Social Recommendation Widget

Sten Govaerts[1] **Sandy El Helou**[2] **Erik Duval**[3] **Denis Gillet**[4]

Dept. Computer Science[1,3]
Katholieke Universiteit Leuven
Celestijnenlaan 200A, Heverlee, Belgium
{sten.govaerts[1], erik.duval[3]}@cs.kuleuven.be

REACT group[2,4]
Ecole Polytechnique Fédérale de Lausanne
1015 Lausanne, Switzerland
{sandy.elhelou[2], denis.gillet[4]}@epfl.ch

ABSTRACT
This paper presents a federated search and social recommendation widget. It describes the widget's interface and the underlying social recommendation engine. A preliminary evaluation of the widget's usability and usefulness involving 15 subjects is also discussed. The evaluation helped identify usability problems that will be addressed prior to the widget's usage in a real learning context.

Author Keywords
federated search, social recommendations, widget, Web, personal learning environment (PLE), Pagerank, social media, Web 2.0

ACM Classification Keywords
H.3.3 Information Storage and Retrieval: Information search and retrieval—*Information filtering*

INTRODUCTION
Under the umbrella of the ROLE[1] (Response Open Learning Environments) EU FP7 project, which focuses on the use of ICT for lifelong learning and the development of personal learning environments (PLE), a federated search and recommendation widget is developed. PLEs are user customizable learning environments that support re-use, creation and mashups of tools, resources, learning activities, peers and experts [1]. A widget is a small web application that embeddable in a web page allowing easy mashups [2]. Federated search allows simultaneous search over multiple repositories.

Nowadays, a search engine is the browser home page for most people and many even think of the browser and search engine as one[2]. General purpose search engines (e.g. Google) are very useful, but their generality can sometimes be an obstacle and this can make it difficult to know where to search for the needed information [2]. Federated searching over collections of topic-specific repositories can assist with this and save time. When the user sends a search request, the federated search widget collects relevant resources from different social media sites [3], repositories and search engines. Before rendering aggregated results, the widget calls a personalized social recommendation service deemed as crucial in helping users select relevant resources especially in our information overload age [4, 5]. The recommendation service ranks these resources according to their global popularity and most importantly their popularity within the social network of the target user. Ranks are computed by exploiting attention metadata and social networks in the widget. Attention metadata captures user interactions, more specifically liking/disliking/sharing a resource [6].

The widget developed can be exploited in formal learning environments and can as well support non-formal learning. The widget aggregates and recommends heterogeneous resources based on their usage by people sharing the same learning context. Recommended resources ranging from wiki pages, videos, and presentations can be saved, shared, assessed, and re-purposed [1] according to each user's interest. The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 describes the software architecture and the user interface design of the federated search widget. Section 4 presents the recommendation service used by the widget to rank aggregated resources. Section 5 discusses the evaluation methodology and the results of a preliminary study conducted with 16 computer science and engineering students to assess the usability and usefulness of the federated search and social recommendation widget presented in this paper. Section 6 concludes the paper.

RELATED WORK
Since the recent social web 2.0 boom, search engines are trying to anticipate this by including results from the user's social network and by adding other social features [7]. Last year, Google and Microsoft Bing started adding social features to their search. Google experimented in 2008 with Google SearchWiki, which allowed users to re-rank, annotate and share search results. After discontinuation, one social feature of SearchWiki is still available, i.e. the 'star' icon, which promotes a result. Google takes one's personal votes into account without exploring social votes. Still, they

[1]The ROLE project, `http://www.role-project.eu`
[2]What is a browser?, `http://www.youtube.com/watch?v=o4MwTvtyrUQ`

Submitted for review to CHI 2009.

include search results from content on your personal social networks (e.g. Twitter and Google Buzz). Bing collaborates with Facebook to provide similar functionalities: searching in the profiles of your friends and in content liked by your friends. IBM also added social features based on its employee directory, tags, bookmarking behavior and ratings to its intranet to show more relevant blogs, wikis, forums and news to great success [2, 8]. Social search is also a popular research topic. Haystaks [9] extends mainstream search engines with new features: users can create sharable search result folders and the content of these folders is used for recommendations. I-SPY [10] allows people to create search communities and based on the queries and results in these communities it will adaptively re-rank the search results.

To enable search over multiple repositories, one can apply federated or harvested search, which collects all the metadata from all repositories in a central repository for faster retrieval [11]. When working with vast repositories of web 2.0 sources (e.g. YouTube), it is impossible to apply harvested search. As far as federated search is concerned, Ariadne [11] focuses on the interoperability between different repositories. The search engine relies on the Standard Query Language (SQI) to ensure interoperability and to offer a transparent search over a network of repositories. Ariadne also provides harvesting. This is applied in the GLOBE network with 13 repositories[3]. MetaLib and WebFeat provide federated search over scientific content [12]. WebFeat sends the query to all search engines and then shows the results in all the native user interfaces. In contrast with MetaLib that uses it's own UI and communicates with the repositories over the standardized Z39.50 protocol. ObjectSpot [13] is originally a federated search widget for scientific publications, but it is now extended for web 2.0 sources. It uses a cover density algorithm to rank the search results, which can also be manually re-ordered. Basic recommendations based on the user selection of search results are also provided. Extending ObjectSpot with social features was not trivial due to the use of incompatible technology. None of these federated search engines provides social features or social search results. Just as mainstream search engines are exploring social networks and attention metadata (voting & sharing), we adopted this strategy in our federated search engine.

THE DESIGN OF THE WIDGET
In this section, the software architecture behind the search widget is explained and the user interface (UI) design and implementation is discussed.

Software architecture
To enable search over multiple data sources, we employ a client-server architecture, as shown in Figure 1. When the user enters a search term, it is sent to the federated search service (step 1), which transmits it to all the different data sources concurrently (step 2). Currently it queries YouTube, SlideShare.net, Wikipedia, GLOBE and the OpenScout repository[4], but extra sources can be easily added to support different learning scenarios. When all the results are returned from the data sources (step 3), the federated search service re-ranks the results (step 4) based on the metadata with the Apache Lucene library [14]. The ranked results are returned to the widget in the ATOM format [15] (step 5), which enables us in the future to make the service OpenSearch compliant [16]. OpenSearch allows search engines to publish search results in a standard and open format to enable syndication and aggregation. In future, we plan to adapt the service to return results every time the repositories return them to improve search speed. Once the widget receives the search results, they are presented to the user and the search result URLs are sent to the recommendation service (step 6). This service will return recommendations, based on the attention metadata stored in the database. The recommendations are sent back to the widget, where they will be presented to the user. The user can interact with the search results, e.g. preview a movie inside the widget or (dis)like it. When some of these interactions happen, they are tracked and the attention metadata (basically the user, the URL of the search result and the action) is sent to the recommendation service (step 7). The service then stores the attention metadata in a database (step 8) to be able to calculate recommendations later. The next section describes the recommendation algorithm in more detail. The client-server architecture enables us to expose repositories not openly accessible by deploying the service inside the intranet.

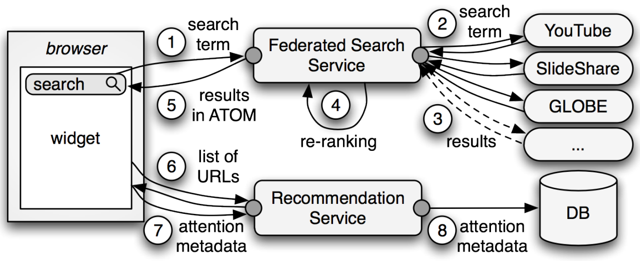


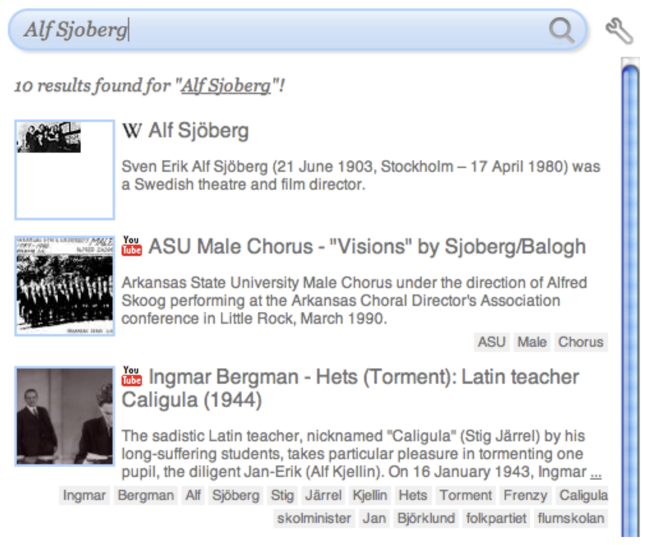
Figure 1. The client-server architecture of the federated search and recommendation service.

User Interface Design
The main design goal was to provide a simple, clean search interface with visually rich search results to enable better decision making while selecting a search result. The widget provides a simple Google-like search interface over multiple web 2.0 data sources. Although advanced search settings are available (see Figure 3), they are not visible by default. Morville et al. [2] advice this as well, because advanced search is often used by expert users. Figure 3 shows the advanced search settings where the wanted media types, repositories and social recommendations can be configured. This can be operated with the wrench icon.

The search results are presented in a uniform way (see Figure 2): basic metadata and tags together with a screenshot

[3]The Global Learning Objects Brokered Exchange (GLOBE) alliance, `http://www.globe-info.org`

[4]The OpenScout repository, `http://www.openscout.net/demo`


Alf Sjoberg

10 results found for "Alf Sjoberg"!

W Alf Sjöberg

Sven Erik Alf Sjöberg (21 June 1903, Stockholm – 17 April 1980) was a Swedish theatre and film director.

ASU Male Chorus - "Visions" by Sjoberg/Balogh

Arkansas State University Male Chorus under the direction of Alfred Skoog performing at the Arkansas Choral Director's Association conference in Little Rock, March 1990.

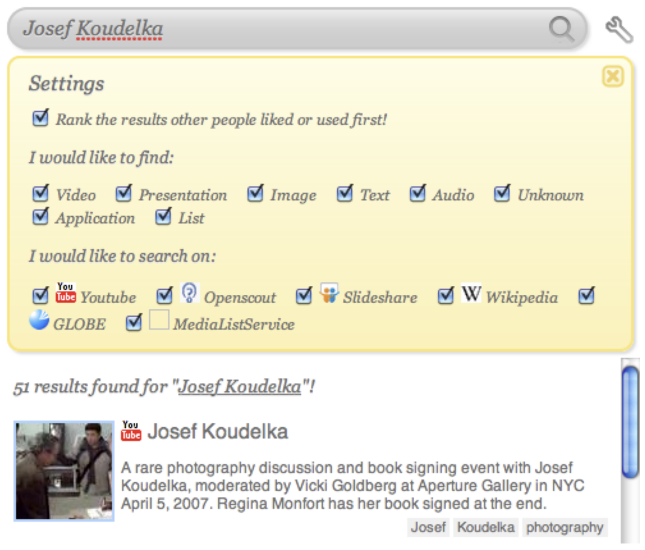
ASU Male Chorus

Ingmar Bergman - Hets (Torment): Latin teacher Caligula (1944)

The sadistic Latin teacher, nicknamed "Caligula" (Stig Järrel) by his long-suffering students, takes particular pleasure in tormenting one pupil, the diligent Jan-Erik (Alf Kjellin). On 16 January 1943, Ingmar ...

Ingmar Bergman Alf Sjöberg Stig Järrel Kjellin Hets Torment Frenzy Caligula skolminister Jan Björklund folkpartiet flumskolan

Figure 2. The user interface.


Josef Koudelka

Settings

✓ Rank the results other people liked or used first!

I would like to find:

✓ Video ✓ Presentation ✓ Image ✓ Text ✓ Audio ✓ Unknown ✓ Application ✓ List

I would like to search on:

✓ Youtube ✓ Openscout ✓ Slideshare ✓ W Wikipedia ✓ GLOBE ✓ MediaListService

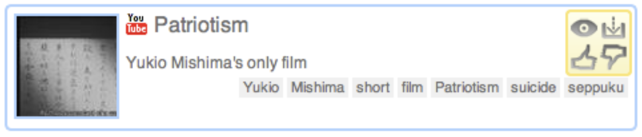
51 results found for "Josef Koudelka"!

Josef Koudelka

A rare photography discussion and book signing event with Josef Koudelka, moderated by Vicki Goldberg at Aperture Gallery in NYC April 5, 2007. Regina Monfort has her book signed at the end.

Josef Koudelka photography

Figure 3. The advanced search settings of the federated search.


Patriotism

Yukio Mishima's only film

Yukio Mishima short film Patriotism suicide seppuku

Figure 4. The taskbar on an actionable result.

if available for easier selection of search results. A small icon next to the title indicates the data source. When the user clicks on a tag, the tag name will be added to the search query to refine the search. This enables two behavioral search patterns: narrowing and pearl growing [2]. Narrowing the search means that when there are too many results, the user can try to refine the search term to find more specific results. Pearl growing is when the user finds one good result and then mines its content and metadata to refine the search.

Ideally the user can complete the necessary tasks while searching without disrupting the search experience or leaving the application. To achieve this, a task bar is shown in the top right corner of every search result when the mouse cursor is hovered over (see Figure 4). This kind of functionality is called actionable results [2]. Four different actions are available. If it is multimedia, it can be previewed with the 'eye' icon inside the widget to check if it is the wanted result. The user can vote on the results with the thumbs up and down to indicate his preference and taste. This is used for the recommendation engine. He can also select the result and embed it in the widget container for later use, copy the URL or share it on wide range of web 2.0 social platforms. The embed functionality enables the user to save search results. Currently we have this functionality working in Graaasp[5] [17].

RECOMMENDATION SERVICE

The federated search is augmented with a recommendation service, which ranks relevant resources based on whether they were 'liked', 'shared' or 'disliked' by peers. Peers can consist of the target user's friends (retrieved from the OpenSocial API [18]) or the people who are using the tool in the same context (e.g. classmates or colleagues). As resources are being liked, disliked, and shared on different sites by peers, a background ranking process runs regularly in order to add newly selected resources or update the rank of existing ones based on user behavior. The ranking algorithm consists of a random walk applied on a graph connecting people and resources and ranking the latter according to their local popularity. The algorithm takes its root from the original Pagerank algorithm [19] used to rank Web pages according to their popularity. The key idea of Pagerank can be formulated as follows: A node is important if and only if many other important nodes point to it. In other words, if the owner of a Web page *j* refers to a page *i*, he/she is implicitly indicating that page *i* is important or 'authoritative'. It follows that the more incoming links a page *i* has, the more it is considered as globally important because many pages are linking to it. In addition, if authors of 'authoritative' pages link in their turn to other pages, then they also confer importance to the latter. The iterative probability equation that translates the algorithm's key idea is described hereafter. A node's conferred importance is divided equally among all the nodes it points to. Let N denote the total number of nodes or Web pages, OutDegree(*j*) the total number of outgoing links from a page (or node) *j*. A transition matrix $T(N \times N)$ is defined such that, each entry T_{ij} is equal to $\frac{1}{Outdegree(j)}$ if *j* points to *i*, and 0 otherwise. Dangling pages are pages with no outgoing links and so they do not confer any importance to other nodes. To guarantee convergence, they are considered to link to all nodes in the graph with an equal probability. For that, a matrix $D(N \times N)$ is defined such that all entries are 0 except for the dangling columns where entries are equal $\frac{1}{N}$. A random jump parameter λ is introduced to avoid situations where nodes of a graph component form an importance 'sink'. λ defines the probability of randomly

[5]Graaasp, `http://graaasp.epfl.ch`

falling on a page, and ensures that no page will have a zero rank and that every page is reachable from any other one. The damping factor d represents the probability to follow page links instead of jumping on a random page. Starting with an equal rank of $\frac{1}{N}$ to all nodes, the probability equation of landing on a node *i* (or rank of a Web page *i*) at each iteration given the ranks of the previous iteration *k*, is given by:

$$p_i^{k+1} = \frac{\lambda}{N} + d\sum_{j=0}^{N}(T_{ij} + D_{ij})p_j^k \quad \text{with } \lambda, d > 0; \lambda + d = 1 \tag{1}$$

Equation 1 can be understood as a Markov chain where states are pages and the transition between states depends on the link structure of the Web. It can be interpreted as the probability for a random surfer to land on a page or node *i* starting at any node with an equal prior probability, following random links with a probability of *d*, and randomly jumping on a page with a probability of λ. It is worth noting, that since the damping factor *d* is less than 1, the further the nodes are from one another, the less influence they will have on each other's rank.

Prior to applying it in the context of the widget, the original algorithm was subject to two main extensions as in [3] before applying it in the context of the federated search and social recommendation widget. Unlike Web pages that are connected with one single relation type, people and resource nodes in the widget can link to one another via different relation types; some being positive and some negative. In fact, there are three possible relation types going from a person to a resource: liking, disliking, and saving resources. People could be connected together with a bidirectional relation when they are 'friends'. In order to take into account the existence of different relation types with potentially different importance weights, the original algorithm is modified as follows. The complete multi-relational network is viewed as a combination of separate sub-networks each connecting nodes with one specific edge or relation type. Let *E* denote the set of all types of edges. An inner transition matrix $T^e(N \times N)$ and a corresponding weight w_e are defined for each edge type $e \in E$, where w_e is interpreted as the probability for a target actor to follow links within the sub-network *e*, or in other words fall on nodes connected by relations of type *e*. Nodes that do not have outgoing links within a sub-network (locally dangling nodes) are considered as linking to all nodes in the sub-network with an equal probability. For that, a matrix $D^e(N \times N)$ is defined for each type of relation e such that all entries are 0 except for the dangling node columns where entries equal $\frac{1}{N}$. Then, the iterative stationary probability equation of landing on a node *i*, is given by:

$$p_i^{k+1} = \frac{\lambda}{N} + d\sum_{e\in E}\left(w_e\sum_{j=0}^{N}(T_{ij}^e + D_{ij}^e)p_j^k\right) \quad \text{with } \sum_{e\in E} w_e = 1; \lambda > 0; \lambda + d = 1 \tag{2}$$

The transition matrix T^e is defined depending on the type of relation it corresponds to. When it comes to relations representing one-time events such as 'liking' or 'disliking' a resource, T^e is similar to the transition matrix of the original Pagerank algorithm. Let OutDegree$_e$(*j*) be defined as the number of edges of type *e* outgoing from *j*, then the entry T_{ij}^e between *i* and *j* can be written as:

$$T_{ij}^e = \begin{Bmatrix} \frac{1}{\text{Outdegree}_e(j)}, & \text{if } j \text{ points to } i \\ 0, & \text{otherwise} \end{Bmatrix}$$

Second, the original Pagerank algorithm is personalized by valuing the opinion of the target user's peers more than others. Borrowing from White et al. [20], Pagerank is extended to rank nodes according to their relative importance to a root set of nodes. For that, the initial probability equation is changed in such a way that the random surfer starts at the root set with adequate prior probabilities, follows links with a probability of *d*, jumps to random nodes with a probability of λ, and goes back to the root set with a probability of β (where it restarts again). This change results in a bias towards the root set and the nodes strongly connected with it (because of the iterative process). During their experiment, authors used a value of 0.3 for β while acknowledging that the choice is inherently subjective and dependent upon the objective, nature and structure of the graphs considered. Let *u* denote the target actor's node and p_u a variable defined such that it is 0 for all nodes except *u*. Then, the complete iterative stationary probability equation of landing on node *i* is given by:

$$p_i^{k+1} = \frac{\lambda}{N} + \beta p_u + d\sum_{e\in E}\left(w_e\sum_{j=0}^{N}(T_{ij}^e + D_{ij}^e)p_j^k\right)$$
$$\text{with } \sum_{e\in E} w_e = 1; \lambda, d, \beta > 0; \lambda + d + \beta = 1; \tag{3}$$
$$p_u = \begin{Bmatrix} 1, & \text{if } i = u \\ 0, & \text{otherwise} \end{Bmatrix}$$

The Equation 3 can be interpreted as the probability to fall on a node in the graph, starting at the target user (different types of links with a probability of *d* (each with a probability of w_e), jumping to random nodes with a probability of λ, jumping back to the target actor with a probability of β, (and restarting again).

With this in mind, Pagerank's key idea can be reformulated as follows: a node is relatively important to the target user's node if and only if many important nodes connected to this target user via important relation types, positively point to it. The node looses its importance if many important nodes connected to the target node, via important relation types, negatively point to it.

A background process runs regularly to execute two independent random walks and combine their results, after they both converge and yield to a unique ranking of people and resources as explained earlier. On the one hand, one random walk takes into account 'like' and 'save' actions as well as peer relationships. This random walk returns a ranked list of resources with a 'promoting' rank representing the likelihood that the target user will like them. On the other hand, the second random walk takes 'dislike' and peer-relationships into account, and returns a list of resources with a 'demoting' rank representing the probability that the target user won't like them. Relations considered in each random walk as treated as equally important. For instance, liking or saving a resource are given the same importance weight while computing promoting ranks. Once the two random walks converge, the demoting rank is deduced from the promoting one and results are saved to serve the federated search engine. As an example, a resource that receives a promoting rank of 0.7 and a demoting rank of 0.1, will achieve a final rank of 0.6. The final ranks are updated every time the background process runs to take into account new user actions. It is worth noting that the two random walks run independently and are both guaranteed to converge to a unique solution [3]. When a user sends a request, the federated search retrieves a list of relevant resources and sends them to the recommendation service. The latter checks whether these resources already exist and returns their actual rank. As resources are being liked, disliked and shared, a request is sent from the widget to save these resources and corresponding actions and take them into account in the next ranking process.

EVALUATION

In the context of the ROLE project, the widget will be used by students of a French language course at Shanghai Jiao Tong University[6]. Prior to its evaluation and actual usage in this learning environment, the widget is evaluated with fifteen engineering and computer science PhD students at K.U. Leuven (9) and EPFL (6). The usability of the search widget is evaluated, together with the user satisfaction, the understanding of the concept of federated search and the usefulness of the recommendations. We now elaborate on the evaluation methodology and setup, then we present the results.

Evaluation methodology and setup

The evaluation consisted of two phases and touched on three aspects: the usability of the UI, the user satisfaction and the usefulness of social recommendation in a search context.

In the first phase, the participants were introduced to the tool and had to execute a task list during an interview with the think-aloud protocol [21]. When 'thinking aloud', users are asked to explain whatever they do, look at and think about while performing the tasks. The results are discussed in regard to the quality components of Jakob Nielsen [22]: learnability (how easy is it to work with the tool for the first time?), efficiency (how quickly can users perform tasks?), memorability (do users still remember how to operate the tool after a period?), errors (how many errors do users make?) and satisfaction. We did not evaluate the memorability due to the short evaluation period.

After the interview, the interviewees were asked to use the widget during 3 to 4 days. Afterwards they filled out an online survey containing open questions and a user satisfaction survey. To measure user satisfaction, we use the System Usability Scale (SUS), which has 10 Likert scale questions and yields among the most reliable results according to Tullis et al. [23]. In addition, we used the Microsoft Desirability Toolkit (MSDT) [22]. The user has to choose all adjectives describing the system out of a list of 118 and then picks the top five and discusses these. Through this explanation session, we can get more detail of the user's experience and opinion on the software.

[6]Shanghai Jiao Tong University, http://www.sjtu.edu.cn/

Table 1. Success rates of the evaluation tasks (max: 15 users).

Task	1st try	2nd try	Fail
Results from different sources	15	0	0
Share movie on Facebook	7	7	1
Vary or restrict media types	12	3	0
Refine the search	10	5	0
Recognize the recommendations	9	6	0

Evaluation results

Usability results

To start the interview, participants were asked how they normally searched for media. All users use Google and nine use YouTube for video.

The user tasks in Table 1 were selected to evaluate the main functionality of the UI and the understanding of federated search and recommendations. Table 1 displays the number of successful students after the first or second try and those that failed on the tasks. First, they had to search and to explain the UI and tell us where the results originated from. Everyone realized that the results came from different sources due to the logos next to the titles. With respect to social features in the UI, 6 out of 15 interviewees immediately noticed the thumbs up/down feature appearing on mouse over. This requires experimenting with other icon sets in the future. All but one were able to share on Facebook and seven thought that the sharing icon was for downloading. Ten people could refine the search by using the tags at the first try and five first looked in different places, e.g. the settings. When users were asked to choose one media type (e.g. videos), an important feature to adapt results to learning styles and user preferences, all but three did go straightaway to the advanced search settings.

With respect to displaying recommendations, 6 people confused the personal recommendation listing and the general results, which are listed in the same column right under recommended items. More focus should be put on recommended items compared to generally relevant ones. People suggested that having a fixed number of recommendations, linking to the normal results from the top of the widget or having a different background for recommendations could help. When asked why these items were recommended, almost half (6/15) guessed that the recommendations came from the thumbs up/down function. Three thought it was content-based. Four guessed based on searching or previewing. Whether the recommendations were based on data captured in the widget or in the repositories themselves, was not clear for four people.

We could improve this and encourage people to vote with more transparent recommendations [24], giving users more insight into why an item has been recommended to them.

Overall the learnability and efficiency are quite satisfying: most people were able to accomplish tasks in one try. Small fixes might improve these numbers even more. No fatal errors occurred during the evaluation.

User satisfaction and usefulness results.
Only 14 of the 15 users (K.U. Leuven: 9, EPFL: 5) were able to participate in the second evaluation session. We inquired more about the usefulness of the recommendations. Eleven confirmed that having recommendations together with search results was useful for them and they also preferred the recommendations separately at the top of the result list. To measure the quality of the recommendations, we compared our approach to their current most used media search solution, which we learned in the first session: Google. We are definitely not claiming that our approach is better than Google, but we are interested how the social recommendations perform compared to Google. To test this, the participants were split up in two groups and were asked to do two tasks using the widget embedded in iGoogle. iGoogle.com is a personal web portal with OpenSocial functionalities [18]. Every group had to search for a different set of two fixed terms and indicate likes and dislikes. The next day, we switched the search term sets between the groups, so that one group would get the social recommendations of the other group, and everybody had to indicate how many of the first 10 Google results and of the first 10 recommendations in the widget were relevant. This way we can measure the average precision at 10 for Google and our widget.

According to the participants, Google was returning more relevant results (average precision at 10 of Google: 65%; average precision at 10 of our widget 50%). We hoped that the social recommendations would improve the results. One of the main complaints was that the federated search results have less variation than Google's. Most of the top results are always videos from YouTube and the users did not tweak the media types nor the repositories in the settings to modify this. The social recommendations only slightly outperform Google with the query 'stand up comedy' (precision at 10 for Google: 60%; precision at 10 for our widget: 63%), which can also be related to the high concentration of YouTube results and the adequacy of having most YouTube videos for that specific query term. And their statements also confirmed that our widget was better for queries requiring audio and video. The precision could probably be improved by varying recommended results based on their media type and not just their rank instead of counting on the user to manually request different media types.

The user satisfaction is measured with SUS and MSDT surveys. The average SUS score is 66.25%. According to Bangor et al. [25] the median of 3500 SUS surveys is 70%. Due to this lower than expected result, we analyzed the answers on the 10 questions. There is a gap between the scores. The participants of EPFL all score the lowest and only one person of K.U. Leuven has a score below average. To analyze this in more detail over each question instead of just with the average SUS scores, we performed a cluster analysis using both hierarchical and k-means clustering using the Euclidean distance. Both techniques yielded the same result: there is a cluster of all users with the low SUS scores (one K.U. Leuven and all EPFL students) and one with all students with above average SUS scores. The average SUS scores of two clusters are respectively: 49.50% and 75.55%. Due to the geographical distance, two different interviewers did the first evaluation session. Except for regional attitude differences towards rating and because the second evaluation session was identical for everyone, we were only able to link this outcome to differences in the first session. The K.U. Leuven PhD students were using the widget in a blank HTML page, while at EPFL the interview was done using iGoogle. From the open questions in the survey, we also learned that most people did not like the iGoogle experience. Different reasons are stated:

- iGoogle distracts because there are other non related widgets that update their UI or draw attention.
- iGoogle shows multiple widgets next to each other and on a low resolution screen the iGoogle and widget UI can become very dense with information, which reduces the overview and can limit the preview capabilities.
- The same remark was noted with the height.
- The users do not like iGoogle in general.

In a separate interview with the K.U. Leuven student, he said he disliked using iGoogle and made similar remarks on the limited width and height.

Some answers from the SUS test are aligned in both clusters. Everybody agrees that they would not need the support of a technical person while using the widget and do not need to learn a lot before they can start with the tool. This supports the high learnability result from the first session. They also concurred that the functions were well integrated in the tool. Some other general suggestions were also made: faster search would be very welcome and maybe split up the search results in tabs per media type.

Next to the SUS survey, the participants were also asked to rate the user-friendliness of the widget on 7 point Likert scale ('Worst imaginable' - 'Awful' - 'Poor' - 'OK' - 'Good' - 'Excellent' - 'Best imaginable'). On average they answered 'Good' ('OK': 3, 'Good': 7, 'Excellent': 4). A quite remarkable thing was that a person with one of the lowest SUS scores rated the user-friendliness as 'Excellent' and most of the other with low scores as 'Good'. This indicates a clear difference between the user satisfaction and the user-friendliness.

A word cloud, showing the frequency of the selected MSDT adjectives, is presented in Figure 5. There is a wide variety of positive and negative words, but the positive ones predominate. Almost all the negative adjectives are from the people with a low SUS score. Slow is an important remark and

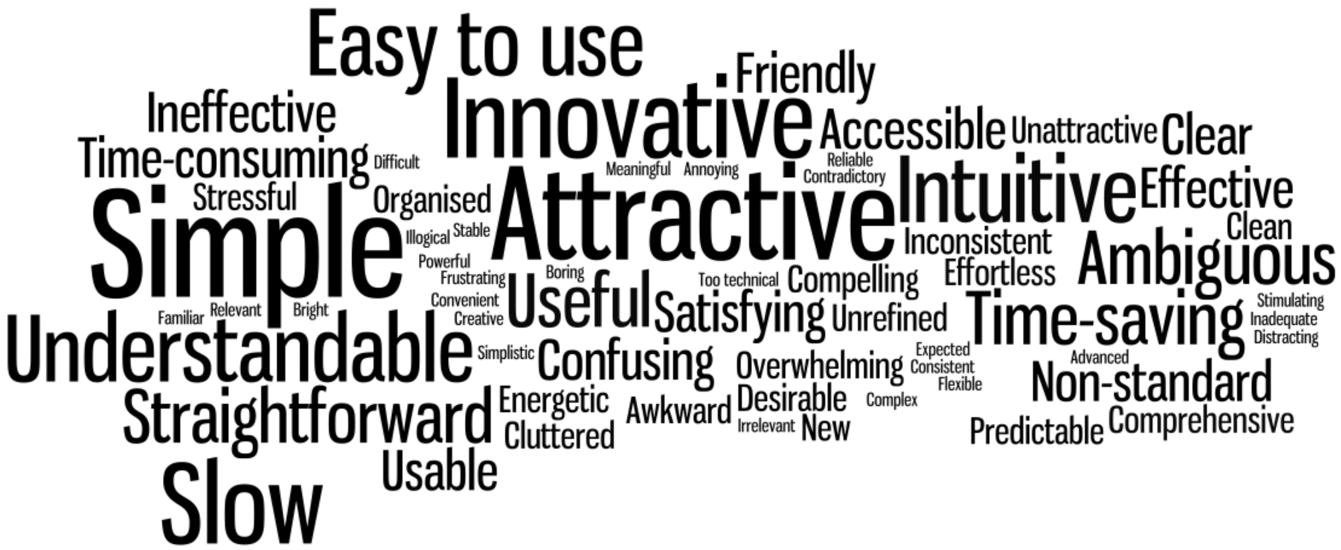


Figure 5. A word cloud based on the frequency of the selected adjectives of the MSDT.

is also related with time-consuming and ineffective, which is due to the current federated search implementation and should be solved by returning the results to the widget once they become available on each repository. Some people indicated in their top 5 that 'stressful', 'confusing' and 'distracting' is related to their experience with iGoogle, because too much information is shown, which could be due to the narrow widget width in iGoogle and in which case the description and tags will take up too much vertical space.

Many of the most selected adjectives are related to high usability: easy to use, simple, intuitive, understandable and straightforward. Some users discussed the usefulness of the recommendations in their top 5 and linked this with 'reliable' and 'usable'.

Thanks to this evaluation we collected a good list for possible improvements, which should be addressed before the evaluation with students from Shanghai Jiao Tong University commences. As this user study was not done with users using a real PLE in a real learning context we plan to re-evaluate the user satisfaction with the Chinese students. This would help to assess the usefulness of peer recommendations in a real-life scenario better than an offline controlled experiment, and with users who actually share the same learning context and interests.

CONCLUSION AND FUTURE WORK

This paper presented a federated search widget with social recommendations for use in PLEs. The social recommendation algorithm is based on an extension of the Pagerank algorithm. We evaluated the usability and user satisfaction of the widget together with a preliminary study of social recommendation usefulness. This study was done as a preparation of a real-world evaluation with Chinese university students.

The overall usability is good and we collected a short list of potential improvements: reconsidering the choice of voting and sharing icons, putting more focus on recommended items compared to other ones, making recommendations transparent, adding more repositories and re-ranking items to ensure media type diversity. The SUS user satisfaction score was 66%, but we discovered a bias in the user evaluations. One group of users rated their user satisfaction much lower based on their experience with iGoogle together with the widget. The user satisfaction needs to be further evaluated in a real-world setting.

An evaluation with students from Shanghai Jiao Tong University to evaluate the usefulness of the widget and the recommendations within a PLE is planned in the coming month. In the near future, there is also an evaluation planned using the widget in a business setting of an international company, FESTO[7]. On the longer term, we plan to realize real-time federated search with support for SQI and be fully OpenSearch complaint to ensure interoperability. Findability can also be improved by for example implementing auto-suggestions for search terms and more visual search paradigms.

Acknowledgement
This research leading to these results has received funding from the European Community Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 231396 (ROLE) and no. 231913 (STELLAR).

REFERENCES

1. S. Downes. Learning Networks in Practice. *Emerging technologies for learning*, 2, 2007.
2. Peter Morville and Jeffery Callender. *Search Patterns: Design for Discovery*. O'Reilly, 1st edition, 2010.
3. Sandy El Helou, Denis Gillet, and Christophe Salzmann. The 3A Personalized, Contextual and

[7]FESTO, `http://www.festo.com`

Relation-based Recommender System. *Journal of Universal Computer Science*, 16(16):2179–2195, 2010.

4. Tiffany Y. Tang and Gordan McCalla. A multidimensional paper recommender: Experiments and evaluations. *IEEE Internet Computing*, 13:34–41, 2009.

5. Il Im and Alexander Hars. Does a one-size recommendation system fit all? the effectiveness of collaborative filtering based recommendation systems across different domains and search modes. *ACM Trans. Inf. Syst.*, 26, November 2007.

6. Martin Wolpers, Jehad Najjar, Katrien Verbert, and Erik Duval. Tracking actual usage: the attention metadata approach. *Educational Technology & Society*, 10(3):106–121, 2007.

7. Brynn M. Evans and Ed H. Chi. Towards a model of understanding social search. In *Proc. of the 2008 ACM conference on Computer supported cooperative work*, CSCW '08, pages 485–494. ACM, 2008.

8. David Carmel, Naama Zwerdling, Ido Guy, Shila Ofek-Koifman, Nadav Har'el, Inbal Ronen, Erel Uziel, Sivan Yogev, and Sergey Chernov. Personalized social search based on the user's social network. In *Proc. of the 18th ACM Conf. on Infor. and Knowledge Management*, CIKM '09, pages 1227–1236, 2009.

9. Barry Smyth, Peter Briggs, Maurice Coyle, and Michael OMahony. Google shared. a case-study in social search. In *User Modeling, Adaptation, and Personalization*, volume 5535 of *LNCS*, pages 283–294. Springer Berlin / Heidelberg, 2009.

10. Jill Freyne and Barry Smyth. An experiment in social search. In *Adaptive Hypermedia and Adaptive Web-Based Systems*, volume 3137 of *LNCS*, pages 95–103. Springer Berlin / Heidelberg, 2004.

11. Stefaan Ternier, Katrien Verbert, Gonzalo Parra, Bram Vandeputte, Joris Klerkx, Erik Duval, Vicente Ordonez, and Xavier Ochoa. The ariadne infrastructure for managing and storing metadata. *IEEE Internet Computing*, 13:18–25, 2009.

12. Xiaotian Chen. Metalib, webfeat, and google: The strengths and weaknesses of federated search engines compared with google. *Online Information Review*, 30(4):413–427, 2006.

13. Zdenek Drbálek, Tomáš Dulík, and Robert Koblischke. Developing components for distributed search engine objectspot. In *Proc. of the 8th WSEAS Int. Conf. on Distance Learning and Web Engineering*, pages 82–85. WSEAS, 2008.

14. Erik Hatcher and Otis Gospodnetic. *Lucene in Action (In Action series)*. Manning Publications Co., 2004.

15. M. Nottingham and R. Sayre. The Atom Syndication Format, RFC 4287.

16. Ralph Le Van. OpenSearch and SRU: Continuum of Searching. *Information Technology and Libraries*, 25(3):151–153.

17. Evgeny Bogdanov, Sandy El Helou, Denis Gillet, Christophe Salzmann, and Stephane Sire, editors. *Graaasp: a web 2.0 research platform for contextual recommendation with aggregated data*. ACM, 2010.

18. J. Mitchell-Wong, R. Kowalczyk, A. Roshelova, B. Joy, and H. Tsai. Opensocial: From social networks to social ecosystem. In *Digital EcoSystems and Technologies Conference, 2007. DEST '07. Inaugural IEEE-IES*, pages 361 –366, 21-23 2007.

19. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.

20. Scott White and Padhraic Smyth. Algorithms for estimating relative importance in networks. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 266–275. ACM, 2003.

21. C. H. Lewis. Using the "thinking aloud" method in cognitive interface design. Technical Report RC-9265, IBM, 1982.

22. Jakob Nielsen. *Usability Engineering*. Morgan Kaufmann, 1st edition edition, September 1993.

23. Thomas S. Tullis and Jacqueline N. Stetson. A comparison of questionnaires for assessing website usability. In *Usability Professionals' Association Conference, UPA*, 2004.

24. Kirsten Swearingen and Rashmi Sinha. Interaction Design for Recommender Systems. *Designing Interactive Systems*, 2002.

25. Aaron Bangor, Philip Kortum, and James Miller. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of Usability Studies*, 4(3):114–123, May 2009.