Directing and Combining Multiple Queries for Exploratory Search by Visual Interactive Intent Modeling

Jonathan Strahl, Jaakko Peltonen, Patrik Floréen

▶ **To cite this version:**

Jonathan Strahl, Jaakko Peltonen, Patrik Floréen. Directing and Combining Multiple Queries for Exploratory Search by Visual Interactive Intent Modeling. 18th IFIP Conference on Human-Computer Interaction (INTERACT), Aug 2021, Bari, Italy. pp.514-535, 10.1007/978-3-030-85613-7_34 . hal-04292383

HAL Id: hal-04292383

https://inria.hal.science/hal-04292383

Submitted on 17 Nov 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

This document is the original author manuscript of a paper submitted to an IFIP conference proceedings or other IFIP publication by Springer Nature. As such, there may be some differences in the official published version of the paper. Such differences, if any, are usually due to reformatting during preparation for publication or minor corrections made by the author(s) during final proofreading of the publication manuscript.

Directing and Combining Multiple Queries for Exploratory Search by Visual Interactive Intent Modeling*

Jonathan Strahl[1][0000−0002−7818−5178], Jaakko Peltonen[2][0000−0003−3485−8585], and Patrik Floréen[3][0000−0001−7347−0685]

[1] Department of Computer Science, Aalto University, Finland
[2] Faculty of Information Technology and Communication Sciences, Tampere University, Finland
[3] Dept. Computer Science, University of Helsinki & Aalto University, Finland
jonathan.strahl@aalto.fi, jaakko.peltonen@tuni.fi, patrik.floreen@helsinki.fi

Abstract. In interactive information-seeking, a user often performs many interrelated queries and interactions covering multiple aspects of a broad topic of interest. Especially in difficult information-seeking tasks the user may need to find what is in common among such multiple aspects. Therefore, the user may need to compare and combine results across queries. While methods to combine queries or rankings have been proposed, little attention has been paid to interactive support for combining multiple queries in exploratory search. We introduce an interactive information retrieval system for exploratory search with multiple simultaneous search queries that can be combined. The user is able to direct search in the multiple queries, and combine queries by two operations: intersection and difference, which reveal what is relevant to the user intent of two queries, and what is relevant to one but not the other. Search is directed by relevance feedback on visualized user intent models of each query. Operations on queries act directly on the intent models inferring a combined user intent model. Each combination yields a new result (ranking) and acts as a new search that can be interactively directed and further combined. User experiments on difficult information-seeking tasks show that our novel system with query operations yields more relevant top-ranked documents in a shorter time than a baseline multiple-query system.

Keywords: interactive information retrieval · information visualization · exploratory search · intent modeling · query combination.

1 Introduction and Related Work

Information seeking is an everyday task in domains ranging from internet pages to specialized corpora such as scientific publications, healthcare information and

* Work supported by Academy of Finland (FCAI flagship and grants 313748 & 327352), Business Finland (grants 2115754 & 211548), and Aalto Science-IT.

legal documents. Simple lookup search is not sufficient for all information seeking: studies estimated up to 50% of searching is informational and the search behavior is exploratory, spreading over multiple queries [9,16]. For example, academic search is exploratory, uncertain, multifaceted and successive [33]: an average scientific search session requires 15 queries [14]. Exploratory search is related to the concept of recommender systems: both help the user find new items of interest. Exploratory search benefits from active engagement with the user, through novel user interfaces and relevance feedback [21]. Exploratory search is hard as it can be difficult for the user to formulate appropriate queries for their information need [40], especially in standard search systems that only support typed queries and show results as document lists; if a query does not yield good results it can take much cognitive effort to explore query reformulations [32].

Methods have been proposed supporting exploratory search [11,12,25,37,42]. Early work on adaptive search [5] and adaptive visualization [1] noted search results improved when the system adapted to the user. Correspondingly, in context-aware search [41] explicit or inferred context can be used to adapt results. In search the user's information need is also called "intent", and search systems must model it from limited data [29]. A dashboard visualizing document similarity was proposed [20], allowing relevance feedback to documents or terms but without visualizing term relationships, similar to a baseline system in [29].

Searching by building Boolean queries has been supported by interactive visualization of Boolean expressions [31], but without recommendation and needing manual construction. Further, the Cluster Map [15,39] can create a graph of document subgroups sharing particular ontology terms such as keywords. The user can select such keywords; documents having them are then shown in clusters. The user can combine keywords with logical "AND" to find documents with intersecting keywords. However, there is no modeling of user intent, users must go through a complete keyword list to select relevant ones, and documents must explicitly contain selected keywords or their combinations to be shown which may rule out relevant documents; in contrast, we will propose a system that models user intent, infers and presents recommended relevant keywords, and infers a probabilistic document ranking more tolerant to variability of keyword use.

A system for exploratory search building a probabilistic "interactive intent model" of the user's search intent was shown in [29], and extended to allow negative feedback in [27]. We will propose a system which supports exploratory search with multiple simultaneously shown queries which we call "search streams", each with recommendation of content based on probabilistic models of user intent, and allows combing streams by operations between the models themselves.

In the "interactive intent modeling" approach [27,29,30], a radial keyword visualization communicates the beliefs of the intent model with the user: predicted relevant keywords are shown according to the underlying user intent model and organized as directions in an information space. Radial visualization has proven beneficial also in other work [24] not featuring user modeling or keyword recommendation, but in the interactive intent modeling approach the visualization directly shares the beliefs of the model with the user [29], so the user can learn

about the search space and provide relevance feedback to direct the search. We propose a system where radial visualization is used over multiple search streams.

In exploratory search, web search logs revealed users open several tabs [17,18] and work on them concurrently [7]. In academic search tasks users use multitasking with parallel and recurrent reformulation [29]. Such results motivate making multiple searches available in the same view. Andolina et al. [3] showed benefits of letting the user work on multiple exploratory searches with intent modeling in parallel, where each user search query and related visualization is denoted as a "stream". They showed side-by-side streams and allowed keyword dragging from one stream to another. This requires each keyword to be transferred separately and assigned a suitable relevance weight, and the result does not yield an intersection of the information needs; our proposed approach is much simpler for the user, combining two entire streams in two mouse clicks.

A challenge for exploratory search and information retrieval are difficult queries [10,22], where few or none of the top ranked documents are relevant. A recent work [27] showed interactive intent modeling [29] could resolve many difficult queries, but in some queries positive relevance feedback did not sufficiently improve relevance of top documents; adding negative relevance feedback improved such search results. Our new system further improves performance on difficult queries; an example is seeking documents that should contain multiple subtopics, e.g. "machine learning" and "mobile phones" in each document.

Combining evidence from multiple queries [2,6,13] and combining data sources for a query [35] are shown to improve results, motivating a system to aid in combining queries. However, interactive systems for combining evidence are missing. Some work on exploratory search uses interactive visualizations [3,20,23,27,29,30], but without combinations of search streams, which we emphasize in this work.

We address a crucial need of users working with multiple exploratory search streams: a user may want to find what is in common between two searches, and what is the difference between them. In the previous example, given two queries "machine learning" and "mobile phones" a user may want to find documents relevant to both queries (both "machine learning" and "mobile phones"), or documents relevant to one but not the other (relevant to "machine learning" but not to "mobile phones"). Simple set-theoretic operations on queries or on document sets would not suffice for these needs; for example, a query with the concatenated query terms can return documents with keywords in only one set, but we need documents having keywords in both sets. To solve the problem, we introduce two operations between pairs of streams: **intersection** and **difference**. Intersection lets the user see what relevant content is in common between two search streams, and difference finds content relevant to one stream but not the other. They are inspired by set-theoretic analogues but are novel **operations on search intent models themselves**. The intent-model based operations result in novel probabilistic relevance-scoring methods. Both operations support exploration: they produce a new stream the user can interact with.

In this work we create the first exploratory visual system supporting operations on intent models. We provide a mathematically well-grounded solution for

such operations. We support the user to visually explore the result, and direct the search, through a new visualized intent model. Our main contributions are:

1. **We address challenging exploratory search** where simple lookup search or naive Boolean operations on document lists/keyword filters are not enough. The user seeks documents relevant to an underlying information need rather than initial keywords; the user must find documents beyond initial ones and novel keywords leading to them. Our interactive intent modeling uses parallel search streams, probabilistic user intent modeling, operations on intent models, and interactive visualization for relevance feedback; through the models even relevant papers not having any original keywords can be found.
2. **Multiple streams**. Previous work [27,29,30] only considers individual streams. We extend the single-stream system in [27], allowing positive and negative feedback, to multiple streams shown side-by-side in one display. Each stream has its own intent model and result list; the user can interact with all streams.
3. **Novel operations.** We add novel interaction between these streams: operations that combine streams as intersections and differences. These operations go beyond simple set-theoretic Booleans of document list and keyword filters [31]: they operate on probabilistic intent models.
4. **Novel ranking.** We show that simple mathematics do not suffice for operations on intent models, and develop a new nontrivial solution yielding novel ranking criteria. We prove it works where a simple concatenation would not; and it produces a new intent model that can be further directed by the user.
5. **Exploration of potentially relevant keywords.** This is the first work with multiple streams including both relevant keywords and predicted future relevant keywords. Previous work on multiple streams [3] did not allow visual exploration of potentially relevant keywords.
6. We introduce a **visual interface** for exploring and combining multiple queries. Prior work only supports a single query [27,29,30], lacks operations between queries [3], or only facilitated query construction [31].

We demonstrate our system on exploratory search of scientific articles, although it is applicable to other domains. We compare to Peltonen et al. [27].

2 Directing and Combining Queries

We propose an exploratory search system that allows multiple streams side by side and enables operations between streams. Each stream is based on the UI and model of [27], represented both by a ranked document list and by a visualization of the estimated search intent as relevant keywords organized in a radar-like interactive visualization, see Fig. 1; for an example with multiple streams side by side, see Fig. 2. We now present a walk-through of the interface.

The system in [29,30], further developed in [27], had a query box, a radar interface showing keyword relevance and the resulting list of documents retrieved; we made minor modifications to this design in order to more conveniently display multiple search streams side-by-side and added buttons for implementing

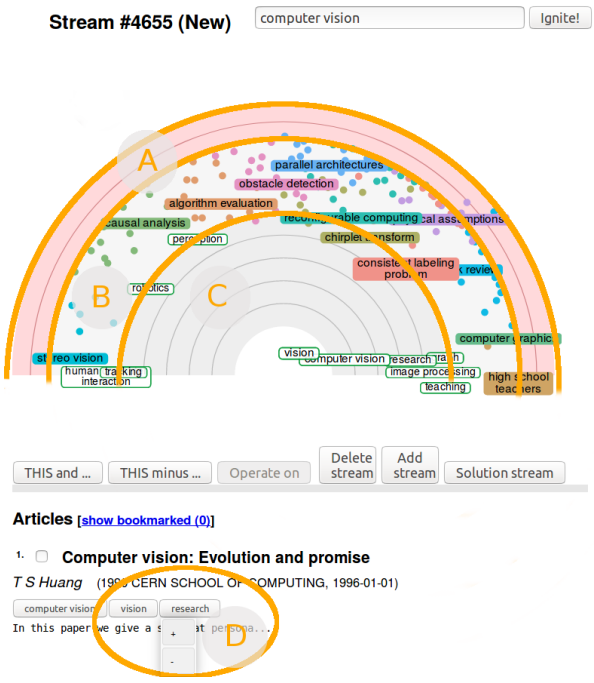

Fig. 1. A single search stream: query box at the top, visualization as a half-circle radar of keywords below, operation buttons below the visualization and article list at the bottom. The ten most relevant keywords are displayed in area C, predicted future relevant keywords in area B, and unwanted keywords in area A when negative feedback is given. Relevance +1 or -1 can be given to keywords in the article list using a pop-up button D. See https://github.com/strahl2e/OpsOnExploratorySearchIntentModels for high-resolution Figures 1-2.

our operations, see Fig. 1. On the radar, the ten keywords estimated to be most relevant are displayed in the inner area (C), with the most relevant closest to the center. The ten most unwanted keywords, i.e, those predicted to be most unwanted based on negative feedback, are in the outer area (A). The intermediate area (B) contains predicted future relevant keywords; the user can hover the mouse over the "bubbles" to enlarge them and show the text following the principle of "overview first, zoom and filter, then details on demand" [34]. Similar keywords are placed at a similar angle on the radar.

The user initializes a search by entering a query and clicking "Ignite" (Fig. 1). This displays the intent visualization and article results for the query.

Feedback on a single search As in [27], a user can drag a keyword on the radar to give feedback: placing it in the center gives maximum relevance 1, in area C relevance drops linearly to 0 moving outwards, in area B it is 0 and in area A it drops linearly from 0 to -1 moving outwards, yielding negative feedback. Feedback +1 or -1 can also be given to a keyword in the document

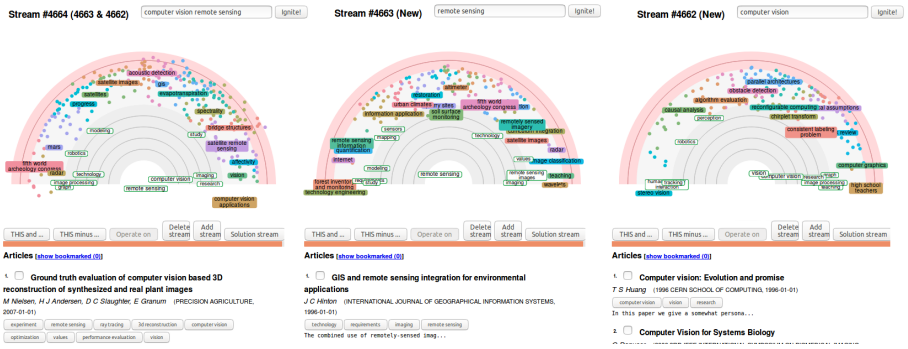

Fig. 2. Result (left stream) of an intersection operation between "computer vision" (right) and "remote sensing" (middle).

list by hovering over it and pressing a + or − pop-up button (D). Feedback can be given to multiple keywords; the search can then be updated by clicking the center of the radar. The search can be refined by repeating this process.

Multiple query streams On pressing the "Add stream" button a new stream appears at the left end of the view. The user can inspect and give feedback to all streams in the view, scrolling left/right if needed. The user can make an intersection or difference operation on any two streams; this results in a new stream to the left of the view. Feedback can be given to the new streams and further intersection and difference operations can be applied to them. Users can also delete streams with the "Delete stream" button.

Stream operations An **intersection** operation between streams A and B is done by pressing the "THIS and ..." button on A and the "Operate on" button on B. This creates a new stream that shows a combination of keywords in A and B and ranks documents based on relevance to both searches, see Fig. 2. A **difference** operation between A and B is done by clicking "THIS minus ..." on A and the "Operate on" on B. This creates a new stream that shows keywords relevant to A but not B, and ranks documents relevant to A but not B .

3 Exploratory Search with Interactive Intent Modeling

An interactive intent model is a probabilistic representation of what content is relevant to the user, parameterized by keywords and their relevance. We describe the model of [27] and our modifications at different phases of exploratory search.

3.1 Initializing the Search Stream

Typing a query initializes the search. We use query likelihood with Jelinek-Mercer (JM) smoothing [43] to rank documents; initial query terms have weight 1. The top N (we use N=100) ranked documents are retrieved. Keyword propagation is then applied: for each unique keyword from the top N documents that

keyword is added to keyword sets of other documents containing it in the text. Instead of keeping the ranking from Lucene as in [27], we rerank the documents based on the propagated keywords, with a modified smoothing, see Sec. 3.4.

3.2 User Feedback

The user can give feedback to keywords to update the search. The intent model, with the accumulated user feedback, estimates relevance for all keywords; the relevance estimates are used to retrieve documents and draw the visualization.

A user feedback is a scalar $r_k \in [-1, 1]$ for a keyword $k \in \{1, 2, ..., M\}$, where M is the number of unique keywords. Each keyword has a TFIDF feature vector $\mathbf{x}_k \in \mathcal{R}^N$ over N documents. Multi-step transition smoothing [27] spreads influence of keywords to related documents: one step is $\mathbf{P} : [\mathbf{P}]_{ij} = \frac{[\mathbf{X}^\top \mathbf{X}]_{ij}}{\sum_{k=1}^{N} [\mathbf{X}^\top \mathbf{X}]_{ik}}$ and the Markov-transition matrix is $\mathbf{P}_{multi} = 0.5\mathbf{I} + 0.25\mathbf{P} + 0.1875\mathbf{P}^2 + 0.00625\mathbf{P}^3$, and the new keyword feature space becomes $\mathbf{X}_{new} = \mathbf{X}\mathbf{P}_{multi}$.

To address the problem of little user feedback, [27] uses exploration-exploitation: predicting relevance scores with upper confidence bounds (UCBs). We addressed a weakness in the LinRel [4] UCB model they used; linear regression can overemphasize uncertain keywords yielding UCB scores above the maximal relevance of 1. It could confuse users if such keywords were shown as top ones instead of those that received even maximal feedback. We instead use Gaussian process (GP) regression [36] so that $\hat{r}^{UCB}(\mathbf{x}) = E_f[f(\mathbf{x})] + \alpha Var_f[f(\mathbf{x})]$, where the expectation and variance are over the GP prior $f \sim GP(0, C(\mathbf{X}_{new}))$ with a radial basis kernel function C, and α controlling the amount of exploration; relevance spreads mostly to similar keywords and uncertain keywords are closer to 0.

3.3 Intent Visualization

The intent visualization represents user intent as keywords on a radial display [26,27]; users can explore the search space and give feedback. The visualization is computed from relevance UCBs (Sec. 3.2) to aid exploration (showing potentially relevant keywords) and exploitation (showing estimated relevant keywords).

For each keyword its relevance UCB is used as its radius. Their angles are organized by similarity: two keywords are considered similar *if their relevance predictions behave similarly with respect to relevance feedback.* A pseudo-feedback (value 1) is added in turn to each of the current inner top ten keywords, to infer new UCBs for all other keywords. The ten pseudo feedbacks thus yield for each keyword ten UCBs, which we collect into a vector representing how relevance of that keyword is affected by user feedback. Two keywords are considered similar if their length-normalized UCB vectors are similar. The neighborhood retrieval visualizer method [38] is used to map the keywords to one-dimensional angles so that similar keywords get neighboring angles on the radial display.

Intent models, UCBs, and optimized angular organization differentiate our display from previous radial search visualizations [24], and the handling of multiple streams and streams representing results of operations with intent models differentiates our display from previous interactive intent visualizations [27,29,30].

3.4 Document Retrieval

Query expansion. With a UCB score for each keyword, the ten keywords with highest UCB scores are added to the existing query weighted by their UCB scores, and are the new inner keywords for the visualization in Sec. 3.3.

Retrieval. Using the reformulated query, the JM unigram language model is used to retrieve a new set of N top-ranking documents from the full collection.

As in [27], a query may include both wanted (positive-weighted) keywords and unwanted (negative-weighted) keywords. Consider a query $\mathcal{Q} = (\mathcal{K}, \mathcal{R})$ with keywords $\mathcal{K} = \{k_1, \ldots, k_R\}$ and their predicted UCB relevance weights $\mathcal{R} = \{r_1, \ldots, r_R\}$. Let us split the keywords into the set of positive keywords $\mathcal{K}^+ = \{k_1, \ldots, k_{R^+}\}$ with weights $\mathcal{R}^+ = \{r_1, \ldots, r_{R^+}\}$ and the set of negative keywords $\mathcal{K}^- = \{v_1, \ldots, v_{R^-}\}$, whose weights $\mathcal{R}^- = \{w_1, \ldots, w_{R^-}\}$ are absolute values of their original negative weights, for convenience of notation. The query likelihood ranking of a document d is based on the likelihood ratio

$$\frac{p(\mathcal{K}^+, \mathcal{R}^+ \mid d)}{p(\mathcal{K}^-, \mathcal{R}^- \mid d)} = \frac{p((k_1, r_1), \ldots, (k_{R^+}, r_{R^+}) \mid d)}{p((v_1, w_1), \ldots, (v_{R^-}, w_{R^-}) \mid d)} = \frac{\prod_{i=1}^{R^+} p(k_i|d)^{r_i}}{\prod_{i=1}^{R^-} p(v_i|d)^{w_i}}, \quad (1)$$

where $p(k_i|d)$ is the probability of generating keyword k_i from document d, $p(v_i|d)$ is the probability of generating unwanted keyword v_i, and the second equality follows because of the independence assumption.

The above principle is used for an original stream; streams created by stream operations require more advanced ranking of documents, described in Sec. 4.

Document reranking. Documents are retrieved from the full collection with a unigram query-likelihood model scoring each document based on the title, keywords and abstract [27]; we rerank documents based on the propagated keywords only. The propagated keywords may be sparse, hence even with JM smoothing the ranking might not tell apart relevant documents; to mitigate the sparsity, we also reward documents for keywords that tend to co-occur with desired keywords. To do so, we introduce Markov transition smoothing, as described in Sec. 3.2, as an extra component of JM smoothing. The score for keyword k_i of query $\mathcal{Q}$ for the jth document from the N retrieved is $p(k_i|d_j) = (1 - \lambda - \beta)\tilde{p}(k_i|d_j) + \lambda p(k_i|C) + \beta[\mathbf{X}_{new}]_{ij}$, where d_j is the jth document, k_i is a query keyword at position i in the set of keywords, $\tilde{p}(k_i|d_j)$ is the keyword frequency in the document divided by document size, $p(k_i|C)$ is the collection probability for keyword k_i that we compute from the N retrieved top-ranking documents, $\lambda \in [0, 1]$ and $\beta \in [0, 1]$ are strengths of the collection smoothing and Markov-transition smoothing respectively: $\lambda + \beta \leq 1$. We do not renormalize $\mathbf{X}_{new}$ per document, which adds a small ranking preference towards documents well-connected in the transition graph.

4 Operations on Exploratory Search Queries

We motivate need for new operation mathematics, show a naive solution fails (Sec. 4.1) and explain **intersection** and **difference** operations (Secs. 4.2 and 4.3).

Simple set-theoretic operations do not suffice for exploratory search. If two search streams were simply finite unordered document sets $\mathcal{D}^A$ and $\mathcal{D}^B$, set-theoretic intersection $\mathcal{D}^A \cap \mathcal{D}^B$ would find all documents in common between the sets, and set-theoretic difference $\mathcal{D}^A \backslash \mathcal{D}^B = \mathcal{D}^A \cap (\mathcal{D}^B)^{\mathsf{c}}$ would exclude all documents in B. However, in search streams this does not suffice:

- Ranks of documents are crucial: a search stream scores and ranks every document, hence the unordered sets $\mathcal{D}^A$ and $\mathcal{D}^B$ both contain the whole corpus (only with different rankings) and rank-unaware set-theoretic operations would do nothing (intersection) or yield an empty set (difference).
- For computational scalability, search systems do not score and rank every document in a corpus, but exclude low-relevance documents early on. Naive intersection of top documents does not suffice: for complicated scientific concepts, it is likely that top documents of any search stream are about that concept only (probabilistic query-based ranking criteria favor such results) so two streams are likely to have no top documents in common.
- Working with documents alone is not enough: in interactive search, each stream must represent a model of the user's search intent, here relevances of keywords, and incorporate user feedback. So operations must also produce a model of intent and its uncertainty to allow further exploration and feedback.

Naive Boolean searches for specific keywords also do not suffice in exploratory search: a hard Boolean requirement, such as "must include machine learning AND mobile phone", would not work to find documents about the underlying concept rather than the specific keywords. Our novelty is enabling operations for this exploratory task, as combinations of intent models and corresponding probabilistic rankings rather than hard Boolean constraints, and integrating them in an interactive system where visualizations suggest related concepts.

4.1 Intersection Operation

The intersection operation aims to find documents relevant to two search streams. Next, we show that a naive query likelihood scoring, resulting from a naive Boolean AND operation of two search queries, is not sufficient to find documents relevant to both searches; then we introduce our model and its scoring.

Naive Scoring Fails for Ranking Intersecting Results Given two queries we wish to rank highly documents that are relevant to both queries. If one simply concatenates the keywords of both queries and ranks them with a query likelihood model, the highest ranking documents may only contain positive keywords from one query. We formalize this problem as follows.

Lemma 1. *When documents are ranked by a query likelihood language model with a bag-of-words document representation, a Boolean AND operation between two queries $\mathcal{Q}^A$ and $\mathcal{Q}^B$ can yield higher scores for documents without intersecting terms than for documents with intersecting terms.*

Proof. The contents of a query are its keywords and their relevances $(\mathcal{K}, \mathcal{R}) = \{(k_i, r_i)\}_{i=1}^{R}$. For simplicity, we assume each query contains only positive keywords so that $(\mathcal{K}^+, \mathcal{R}^+) = (\mathcal{K}, \mathcal{R})$. In a query likelihood model, the query is treated as an observed random event and documents are ranked by the probability to generate the event; two queries $\mathcal{Q}^{\mathcal{A}} = (\mathcal{K}^A, \mathcal{R}^A) = \{(k_i^A, r_i^A)\}_{i=1}^{R^A}$ and $\mathcal{Q}^{\mathcal{B}} = (\mathcal{K}^B, \mathcal{R}^B) = \{(k_i^B, r_i^B)\}_{i=1}^{R^B}$ are treated as two events, and a Boolean AND means both events have happened; the query likelihood for a document is then the probability to generate both events simultaneously from the document.

Let document d contain a set of keywords $\mathcal{T}_d$. For simplicity, assume the probability $p(k|d)$ to generate a keyword k has a uniformly high value ϕ for all $k \in \mathcal{T}_d$ and a uniformly low value γ for other keywords, as would happen with document smoothing. The likelihood ratio (1) has a constant denominator for a query $\mathcal{Q}$ with positive keywords; the numerator becomes the query likelihood

$$\prod_{i=1}^{R} p(k_i|d)^{r_i} = \phi^{(\sum_{i:k_i \in \mathcal{Q} \cap \mathcal{T}_d} r_i)} \cdot \gamma^{(\sum_{i:k_i \in \mathcal{Q} \setminus \mathcal{T}_d} r_i)} .$$

The query likelihood is the probability that the content of query $\mathcal{Q}$ is generated by document d from its language model. The probability to generate two queries $(\mathcal{K}^A, \mathcal{R}^A)$ and $(\mathcal{K}^B, \mathcal{R}^B)$ is the product of their independent probabilities; in the resulting query likelihood, exponents of ϕ and γ can be divided into six terms:

$$\left(\prod_{i=1}^{R^A} p(k_i^A|d)^{r_i^A} \right) \left(\prod_{i=1}^{R^B} p(k_i^B|d)^{r_i^B} \right) = \phi^{r^{A,d} + r^{B,d} + r^{A,B,d}} \cdot \gamma^{r^A + r^B + r^{A,B}} ,$$

where $r^{A,d} = \sum_{i:k_i \in (\mathcal{K}^A \setminus \mathcal{K}^B) \cap \mathcal{T}_d} r_i^A$, $r^{B,d} = \sum_{i:k_i \in (\mathcal{K}^B \setminus \mathcal{K}^A) \cap \mathcal{T}_d} r_i^B$, $r^{A,B,d} = \sum_{i:k_i \in (\mathcal{K}^A \cap \mathcal{K}^B) \cap \mathcal{T}_d} (r_i^A + r_i^B)$, $r^A = \sum_{i:k_i \in (\mathcal{K}^A \setminus \mathcal{K}^B) \setminus \mathcal{T}_d} r_i^A$, $r^B = \sum_{i:k_i \in (\mathcal{K}^B \setminus \mathcal{K}^A) \setminus \mathcal{T}_d} r_i^B$, $r^{A,B} = \sum_{i:k_i \in (\mathcal{K}^A \cap \mathcal{K}^B) \setminus \mathcal{T}_d} (r_i^A + r_i^B)$. Clearly, only the total of the terms in each exponent matters: a document can get a high query likelihood even if it has keywords from one query only, for example several keywords in $(\mathcal{K}^A \setminus \mathcal{K}^B) \cap \mathcal{T}_d$ which yields a high $r^{A,d}$ value, even if the two other terms in the exponent of ϕ are zero. Thus naive query likelihood does not suffice to rank documents by how well they generate terms from two or more sets of terms.

4.2 Ranking for Intersecting Documents

To find documents relevant to two search intent models, and avoid the problem illustrated in Lemma 1, we introduce a novel query likelihood that ranks documents based on the joint probability of generating a keyword from one search stream and a different keyword from another search stream. We consider two queries, each of which may contain both positive and negative keywords, $\mathcal{Q}^A = (\mathcal{K}^{A,+}, \mathcal{R}^{A,+}, \mathcal{K}^{A,-}, \mathcal{R}^{A,-})$ and $\mathcal{Q}^B = (\mathcal{K}^{B,+}, \mathcal{R}^{B,+}, \mathcal{K}^{B,-}, \mathcal{R}^{B,-})$. Our probabilistic ranking for an intersection can be interpreted as a special query likelihood model for the positive keywords: the probability to generate two different keywords such that one is relevant to query A and the other to query B.

This is computed based on several probabilities: the probability $p(k, k' \mid d)$ to generate a pair of keywords k and k' from a document d, the probability $p(k \in A)$ that the first keyword is relevant to stream A, the probability $p(k' \in B)$ that the second keyword is relevant to stream B, and the probability $p(k \neq k')$ that the two keywords are different.

We also want to penalize documents for being able to generate negative (unwanted) keywords of the two queries, i.e., keywords in either $(\mathcal{K}^{A,-}, \mathcal{R}^{A,-})$ or $(\mathcal{K}^{B,-}, \mathcal{R}^{B,-})$. This yields a document score interQL which is a likelihood ratio:

$$interQL(\mathcal{K} \mid d) = \frac{p\,(k \text{ rel. to } A, \text{ distinct } k' \text{ rel. to } B \mid d)}{p\,(v \text{ or } v' \text{ an unwanted keyword} \mid d)} = \frac{\sum_{k,k' \in \mathcal{K}^+} p(k, k' \mid d) p(k \text{ rel. to } A) p(k' \text{ rel. to } B) p(k \neq k')}{\sum_{v,v' \in \mathcal{K}^-} p(v, v' \mid d) p(v \text{ or } v' \text{ unwanted in } A \text{ or } B)} = \frac{\sum_{k,k' \in \mathcal{K}^+} p(k \mid d) p(k' \mid d) r_k^A r_{k'}^B \delta_{k \neq k'}}{2(\sum_{v \in \mathcal{K}^-} p(v \mid d) w_v) - (\sum_{v \in \mathcal{K}^-} p(v \mid d) w_v)^2}\,, \quad (2)$$

where $\mathcal{K}^+ = \mathcal{K}^{A,+} \cup \mathcal{K}^{B,+}$ and $\mathcal{K}^- = \mathcal{K}^{A,-} \cup \mathcal{K}^{B,-}$ are the unions of the positive and negative keyword sets of the two queries, and $\delta_{k \neq k'} = 1$ if $k \neq k'$ and 0 otherwise. We use the relevance values as probabilities that the keywords are relevant: $p(k \text{ rel. to } A) = r_k^A \in [0, 1]$ and similarly for $r_{k'}^B$. In the denominator, either keyword can independently be unwanted, in either query or in both, thus in the result we denote $w_v = p(v \text{ unwanted in } A \text{ or } B) = w_v^A + w_v^B - w_v^A w_v^B$, which reduces to $w_v = \max(w_v^A, w_v^B)$ if the unwanted sets are nonoverlapping.

This probabilistic score solves the issue presented in Lemma 1, as shown next.

Lemma 2. *In an intersection of queries A and B having positive keywords, interQL (2) ranks documents such that documents having intersecting content rank higher than documents without such content.*

Proof. Consider two queries A and B. Assume for simplicity each query contains only positive keywords. Let document d contain a set of keywords $\mathcal{T}_d$. Assume the probability $p(k|d)$ to generate a keyword k has a uniform high value ϕ for all $k \in \mathcal{T}_d$ and a uniform low value γ for other keywords. Then only the numerator of (2) is relevant and becomes

$$\sum_{k,k' \in \mathcal{K}^{A,+} \cup \mathcal{K}^{B,+}} p(k \mid d) p(k' \mid d) r_k^A r_{k'}^B \delta_{k \neq k'} = \sum_{k \in \mathcal{K}^{A,+} \cap \mathcal{T}_d, k' \in \mathcal{K}^{B,+} \cap \mathcal{T}_d} \phi^2 r_k^A r_{k'}^B \delta_{k \neq k'} + \sum_{k \in \mathcal{K}^{A,+} \setminus \mathcal{T}_d, k' \in \mathcal{K}^{B,+} \cap \mathcal{T}_d} \gamma\phi r_k^A r_{k'}^B + \sum_{k \in \mathcal{K}^{A,+} \cap \mathcal{T}_d, k' \in \mathcal{K}^{B,+} \setminus \mathcal{T}_d} \phi\gamma r_k^A r_{k'}^B + \sum_{k \in \mathcal{K}^{A,+} \setminus \mathcal{T}_d, k' \in \mathcal{K}^{B,+} \setminus \mathcal{T}_d} \gamma^2 r_k^A r_{k'}^B \delta_{k \neq k'},$$

where the latter three sums are small as γ is low compared to ϕ; thus the rank of the document is mainly determined by the first sum. That sum is over pairs

(k, k') with $k \neq k'$, such that $k \in \mathcal{K}^{A,+} \cap \mathcal{T}_d$ and $k' \in \mathcal{K}^{B,+} \cap \mathcal{T}_d$, so document d must contain at least one term from $\mathcal{K}^{A,+}$ and a different term from $\mathcal{K}^{B,+}$ to get a nonzero value of this sum. It is then easy to see that such a document scores higher, and thus outranks, another document having zero terms in the first sum, as long as the probability ϕ is sufficiently higher than γ.

Lemma 2 shows our formulation solves the limitation of traditional query likelihood in Lemma 1. While other formulations with such benefit are possible, to our knowledge ours is the first solution and yields good results in experiments.

Ideally we would rank all documents with our probabilistic intersection ranking, but in very large corpuses computational efficiency is key. We thus propose to first use a special unigram query-likelihood based ranking, detailed below, to retrieve a subset of documents, and then rerank them by InterQL. The advantage is that this first-stage retrieval is directly implementable in efficient packages like Apache Lucene, and still yields good enough documents.

First-Stage Retrieval for Intersection by Unigram Query Likelihood We use a special unigram query likelihood retrieval to filter out low-scoring documents as nonrelevant; then rerank the higher-scoring documents with interQL.

Unigram query likelihood retrieval: Keyword weight balancing. We build a unigram query (with JM smoothing in Apache Lucene) from the two original queries for an intersection of streams A and B, with their weighted keywords $(\mathcal{K}^A, \mathcal{R}^A)$ and $(\mathcal{K}^B, \mathcal{R}^B)$, by concatenating the relevance-weighted keywords and balancing the weights so that the resulting L highest ranking documents are maximally relevant to both queries ($L = 100$ in experiments). We found that if we simply use given weights for keywords and multiply the query likelihood ratios for A and B of eq. (1), top ranked documents are often only related to one of the searches, so we apply a search strategy to balance the weight for each search. The unigram query likelihood ratio of (1) is then

$$\left(\prod_{i=1}^{R^Q} p(k_i|d)\right) \left(\frac{\prod_{i=1}^{R^{A+}} p(k_i|d)^{r_i}}{\prod_{i=1}^{R^{A-}} p(v_i|d)^{w_i}}\right)^{\beta} \left(\frac{\prod_{i=1}^{R^{B+}} p(k_i|d)^{r_i}}{\prod_{i=1}^{R^{B-}} p(v_i|d)^{w_i}}\right)^{2-\beta},$$

where the first product is over the R^Q keywords in the concatenated query, other products are over the predicted relevant keywords of A and B, and $\beta \in [0, 2]$ is a balancing factor. An efficient strategy to find a good value of β is to choose a set of values and run them in parallel, then pick the one that maximizes the number of documents having a non-empty positive-set intersection $(\mathcal{K}^{A+} \cap \mathcal{K}^{B+}) \cap \mathcal{T}_d$.

Before the balancing is applied, we: 1) normalize weights in A and B to have the same sum (average of their original sums); 2) multiply weights of keywords in $\mathcal{K}^A \cap \mathcal{K}^B$ by $1/2$ to avoid one shared keyword dominating the score of a document. This is to ensure enough good documents are near the top.

Selecting Keywords from Intersecting Searches *Inner keywords.* For the ten inner keywords on the visualization of a new intersection search we use the

top keywords from each search (top five in the experiments) in the operation. If the two keyword sets have no keywords in common we concatenate the two sets. If the same keyword is in common to both streams, it is added only once, we then add a new keyword from the stream that had higher UCB for the keyword in common, ensuring we get ten unique high-ranking keywords from both streams.

Ranking. In an intersection of searches A and B we use 20 keywords to rank documents according to (2), 10 each from A and B, selected similarly as above.

Removal of "covered" unigram keywords. Keywords with more than one term (e.g. "machine learning") are decomposed into unigrams during efficient retrieval. These introduced unigrams can dominate the intersection ranking due to being more common, but worsen the results due to being generic single words. To solve this, if a unigram and a larger n-gram containing ("covering") it appear in a document, for document ranking we omit the covered unigram.

Feedback on an Intersection. An intersection stream has two underlying intent models. Each inner keyword on the visualization belongs to one of the models; feedback on it goes to that model. Keywords in the intermediate area of the visualization use average UCB values from both models making it unclear which model should get the feedback. For such a keyword, we aim to assign feedback to the intent that the keyword is more related to. We assign the feedback to the model with the smallest uncertainty for the keyword: keywords with small uncertainty in our intent model either already have feedback or are similar to ones with feedback, and are thus likely to be related to the intent of that model.

4.3 Difference Operation

A difference operation returns documents relevant to one stream A and not relevant to the other stream B. Its components are constructed as follows.

Query-text. As it represents desired content, we set it to the query-text of A.

Feedback sets. Differing amounts of feedback in A and B should not make a difference operation overly favor either stream in document ranking or keyword scoring. To ensure an equal role for both streams we take the top-five relevant keywords from A, and give positive implicit feedback to the new stream, and we take the top-five relevant keywords from B, and give negative implicit feedback.

Intent model. With the query and feedback we infer the intent model and learn relevance estimates of all keywords (Sec. 3.2), with TFIDF vectors for keywords computed using the set of $2N$ documents from both original searches.

Given the intent model, we run visualization and retrieval (Secs. 3.3 and 3.4).

5 User Experiments

User experiments investigate the effect of the intersection and difference operations. We compare two versions of the same system, one with our two operations (ours) and the other with them disabled (baseline). The single-stream behavior of both systems corresponds to the state-of-the-art system in [27], modified as

detailed at the end of Sec. 3.4. Comparison between the system with negative feedback [27] and without it [29] is in [27], and between [29] and a common typed query system is in [29]; thus we focus on the effects of our two operations.

***Hypothesis**: With the intersection and difference operations in our system, the user reaches (1) more relevant results (2) more easily and faster.*

5.1 Task Design

The baseline exploratory search system gives support beyond standard queries or keyword-only search [29]. As it is a well-performing system, we do not claim multiple searches and operations between them yield strong benefit in all exploratory tasks — if the user aims to narrow down on one topic, exploration with one stream may suffice; even in some initially difficult queries negative feedback to the stream may yield good results [27]. We thus focus on those difficult queries where such easy fixes do not work, as discussed in Section 1. Recent studies of user-perceived causes of difficult queries revealed multiple subtopics as a main cause [22]. Our intersection operation is designed to rank documents relevant to multiple streams, essentially multiple subtopics, so we focus on difficult queries with multiple subtopics. We explored a number of tasks with multiple subtopics, where the user's goal is to find what is in common between the subtopics. Many could be resolved with single-stream relevance feedback, with enough effort. For the user study we chose three exploratory search tasks that were challenging for the single stream system: 1. "machine learning" and "signal processing", 2. "machine learning" and "mobile phones" and 3. "sparse regression" and "expectation maximization". In each task, the user's goal is to make the top ten ranked documents relevant to the topic of the task, which is a combination of subtopics.

5.2 Experiment Setup

As our experiment is on scientific papers we used computer science university students as a realistic user group. The 17 participants (14 male, 3 female) were PhD (13), MSc (2), and BSc (1) students and 1 postdoc, all experienced search engine users. They were given the three information seeking tasks (Sec. 5.1) to be performed on both systems. We also asked about familiarity about the topics in our tasks. Each user had a HP EliteBook laptop, 24" monitor, mouse, and keyboard. They were shown a ten-minute instructional video on the system, and had five minutes practice on each system before starting the tasks.

We used a balanced within-subjects study design: each user completed all three tasks on both systems. Tasks and systems were ordered with a counterbalanced design to reduce learning bias: for each task essentially equally many users completed it first on our system vs. first on the baseline, and for each task essentially equally many users completed it as the first/second/third task. Users could stop when satisfied; after ten minutes they were requested to finalize and submit their remaining feedback and then stop, ensuring overall experiment time was not excessive. They clicked a button below the stream they considered to have the most relevant top ten documents to denote it as the solution stream.

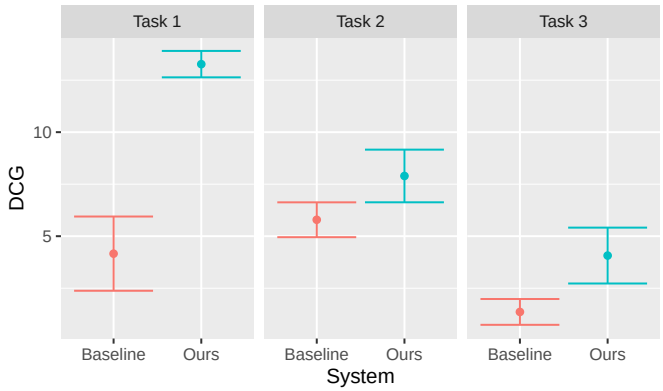

Fig. 3. Discounted cumulative gain (DCG) for tasks 1-3 for the baseline (red) and our system (cyan). Bars are 95% confidence intervals. DCG ranges from 0 (no relevant documents) to 13.6 (all documents certainly relevant).

After completing the tasks, users filled in a user satisfaction questionnaire with 25 questions based on two established frameworks: ResQue [28] and SUS [8].

Data. Both systems used the same data: about 50 million articles from Web of Science, ACM, IEEE, and Springer.

System setup. For the intent model we used an RBF kernel for GP-UCB, where we tuned the width parameter using distances to three nearest neighbors on data samples, and chose a small exploration parameter $\alpha = 0.1$ tuned by testing searches. For the language model (Sec. 3.4), following [43], we set the multi-step Markov transition smoothing parameter and the collection probability smoothing parameter to small values $\beta = 0.05$ and $\lambda = 0.05$.

5.3 Evaluation of Results

Retrieval performance. We collected all the top ten documents from the solution stream of each user for each task on each system, leaving 17 (users) $\times$ 3 (tasks) $\times$ 2 (systems) = 102 result sets of ten documents. For each task, we extracted the list of unique documents from the result sets. Three evaluators (the authors) then gave each unique document a relevance score of 0=certainly not relevant, 1=likely not relevant, 2=likely relevant, 3=certainly relevant without knowledge of which result set(s) the document belonged to. Each evaluator scored independently (average inter-rater correlations in tasks 1-3: 0.75; 0.87; 0.67); the evaluators then discussed discrepancies and came to consensus for each document score. We used discounted cumulative gain (DCG) [19] on the relevance scores of the top ten documents to reward the system for ranking relevant documents higher. Fig. 3 shows the mean DCG score for the top ten documents for each task and system, including the 95% confidence interval, over the 17 users. Our system yields clearly better results than the baseline. The improvement is

Table 1. Counts of operations performed by the users on our system: mean (μ), standard deviation (s.d.), median (med), minimum (min) and maximum (max) computed over all users for each task. The data is reported for all the operations during the experiment (All) and for the subset of operations that resulted in the solution stream (Sol.).

Action	Data	Task	μ (s.d.)	med	min	max
Intersection	All	1	2.2 (1.9)	1	1	6
	Sol.	1	0.9 (0.2)	1	0	1
	All	2	5.8 (2.2)	7	2	9
	Sol.	2	1 (0.4)	1	0	2
	All	3	5.6 (2.3)	6	1	10
	Sol.	3	1.2 (1.0)	1	0	4
Difference	All	1	0 (0)	0	0	0
	Sol.	1	0 (0)	0	0	0
	All	2	0.3 (0.7)	0	0	3
	Sol.	2	0.1 (0.2)	0	0	1
	All	3	0.6 (1.5)	0	0	5
	Sol.	3	0.2 (0.4)	0	0	1

Table 2. Comparison of keyword feedback counts between the baseline (B) and our system (O): mean (μ), standard deviation (s.d.), median (med), minimum (min) and maximum (max) computed over all users for each task. The data is reported for all the feedback during the experiment (All) and for the subset of feedback that resulted in the solution stream (Sol.).

Task	Data	System	μ (s.d.)	med	min	max
1	All	B	20.2 (12.8)	17	3	53
		O	4.5 (6.3)	0	0	18
	Sol.	B	13.6 (14.0)	12	0	47
		O	3.1 (6.2)	0	0	18
2	All	B	29.1 (16.8)	24	5	65
		O	18.2 (11.7)	14	2	39
	Sol.	B	18.1 (21.6)	8	0	65
		O	11.1 (13.0)	6	0	39
3	All	B	23.1 (17.5)	20	2	77
		O	19.1 (17.7)	13	2	68
	Sol.	B	10.9 (9.6)	8	0	34
		O	13.7 (17.6)	8	0	68

statistically significant: repeated measures 2-way ANOVA with users as subjects, performing each task on each system, showed a statistical significance for the system effect (p-value < 0.01). We also verified by another repeated measures 2-way ANOVA that the presentation order of the tasks and systems to the users did not have a statistically significant effect. Thus it is clear from the results that operations gave an overall advantage, especially for task 1.

Task speed. We recorded task durations, and computed the average time for each task on each system. With one-way repeated measures ANOVA tasks 2 and 3 showed no statistically significant difference, but Task 1 had a statistically significant difference (p-value $< 10^{-5}$): average task time was 563 seconds on the baseline and 258 seconds on our system, thus operations made the search faster.

User interaction. Tables 1 and 2 contain statistics of the counts of all stream operations and feedbacks executed by each user. Users took advantage of the operations: all users used operations in all tasks. On average users made several intersection operations for each task, but the difference operation was used only occasionally. Moreover, with the operations users had less need of keyword feedback: they needed clearly less feedback on our system to arrive at their solutions.

Our system yielded results quickly with few operations and little feedback. As shown in Table 1, for task 1 most users needed only one intersection, and as

Table 3. User feedback. Average scores and p-values between the baseline (B) and our system (O), for each question and group of questions (bold font). Responses on a Likert scale: strongly disagree = −2, disagree = −1, neither agree or disagree = 0, agree = 1, strongly agree = 2. We reversed scores for questions where disagree was better (marked with *), so higher numbers are always better. See full table at https://github.com/strahl2e/OpsOnExploratorySearchIntentModels.

Question	**B**	**O**	***p*-value**
I. Quality of Recommended Items	0.2	0.9	$2 \cdot 10^{-6}$
1. The keywords displayed to me matched the search objective	0.5	1.2	$6 \cdot 10^{-4}$
2. The articles displayed to me matched the search objective	0.2	1.2	$2 \cdot 10^{-5}$
3. The system produced good results	-0.2	1.1	$3 \cdot 10^{-6}$
4. The system helps me discover new articles	0.7	1.3	0.03
5. The articles displayed to me are similar to each other*	0.0	-0.2	0.33
II. Interaction Adequacy	-0.4	0.8	$2 \cdot 10^{-5}$
6. The search streams provides an adequate way for me to conduct my search	-0.7	1.0	$3 \cdot 10^{-5}$
7. The keyword feedback provides an adequate way for me to refine my search	0.0	0.6	0.04
IV. Perceived Ease of Use	-0.4	0.8	$8 \cdot 10^{-6}$
10. I easily found the articles that were relevant to my search	-0.5	0.8	$3 \cdot 10^{-6}$
11. It is easy to learn to tell the system what to search for	-0.4	1.1	$3 \cdot 10^{-5}$
12. I feel in control of telling the system what I want	-0.4	0.8	$1 \cdot 10^{-3}$
13. I understood why the articles were recommended to me	-0.1	0.7	$4 \cdot 10^{-3}$
V. Attitude	-0.3	1.0	$1 \cdot 10^{-5}$
14. Overall, I am satisfied with the system	-0.3	1.0	$1 \cdot 10^{-5}$
VI. Behavioral Intentions	0.3	0.7	$5 \cdot 10^{-4}$
15. If a system such as this exists, I would use it to find scientific articles	-0.2	1.1	$3 \cdot 10^{-6}$
16. I think I would use this system frequently if given the opportunity	-0.4	0.6	$2 \cdot 10^{-4}$
21. I thought there was too much inconsistency in the system*	-0.1	0.5	0.01
24. I felt very confident using the system	-0.1	0.4	$7 \cdot 10^{-3}$

shown in Table 2 often without further tuning, to easily get good results outperforming the baseline in speed and result quality. On the baseline, users did not achieve corresponding result quality even with multiple feedback. Tasks 2 and 3 show similar behavior. The only case where users of our system used more feedback than on the baseline was for their solution stream (Sol.) of task 3, but even there the total amount of feedback (All) was smaller on our system than on the baseline. Thus, having operations yields good results with less effort and faster.

Intersections were used both for exploration and for constructing the solution: in all tasks, the total count of intersections (All) in Table 1 is greater than the count for solution streams (Sol.). The difference operation was much less used, but was used in up to three operations per user in task 2 and five in task 3 and contributing to solution streams in those tasks. From interaction data on our system, we found users preferred to make operations first and fine-tune later: a 74% majority of feedback was given to streams produced by an operation.

Serendipity. Unlike manual query construction, intent modeling predicts likely relevant keywords that the user did not directly type, and likely relevant documents having those keywords. This enables finding serendipitous documents: non-trivial relevant documents that are not found by naive filtering for query keywords. Our system allows combining intent models, maintaining the advantage of serendipity in the results. We evaluate serendipity in top ten results for each task and system by counting the total number of serendipitous documents found by the users, i.e., relevant documents that did not contain the query terms of the original queries and would not be found by a logical AND of two queries. Both systems find serendipitous documents; our system finds 55, 72, and 22 serendipitous documents in tasks 1, 2, and 3 respectively, overall better than the baseline, which found 56, 58, and 14.

User satisfaction. In our feedback questionnaire, users were most satisfied with our system. We ran a 1-way repeated measures ANOVA for each question. Table 3 shows the 15 questions with statistically significant difference between systems (p-value < 0.05) out of the 25 questions in total. All significant differences favor our system. Users had better results, keywords, and articles, could conduct and refine the search better, found articles more easily, learned more easily to use the system effectively and felt more in control, were more satisfied, and would use the system and use it more frequently. Thus users had a clearly better experience on our system for these tasks.

The questions in Table 3 were divided into six groups: Quality of Recommended Items (questions 1-5), Interaction Adequacy (6-7), Interface Adequacy (8-9), Perceived Ease of Use (10-13), Attitude (14) and Behavioral Intentions (15-25). Group-wise difference between systems was tested by 2-way repeated measures ANOVA per group, with users as subjects and question and system as factors. The Interface Adequacy group of questions showed no statistically significant difference (thus omitted from Table 3) showing that the addition of the operations did not hurt ability to operate the interface. All other groups had a significant difference, and all significant differences again favor our system.

6 Conclusions and Discussion

We address multiple parallel searches and highlight the complexity of combining user models for exploratory search. As shown in Section 4, simple set-theoretic operations on documents or queries do not suffice. We described a novel system with an interactive visual interface, supporting intersection and difference of search streams. The system allows relevance feedback to the keywords in an interactive visual interface, and involves novel document scoring and keyword exploration models. Our experimental results validate our hypothesis that stream operations improve user performance and satisfaction on difficult queries with multiple subtopics: results were clearly better for all three tasks, average time of task one was clearly lower, and users stated in the questionnaire they found relevant documents more easily. Future work could include experiments on further domains and extension to collaborative search.

References

1. Ahn, J., Brusilovsky, P.: Adaptive visualization for exploratory information retrieval. Information Processing & Management **49**(5), 1139 – 1164 (2013). https://doi.org/10.1016/j.ipm.2013.01.007
2. Anava, Y., Shtok, A., Kurland, O., Rabinovich, E.: A probabilistic fusion framework. In: Proceedings of the 25th ACM International Conference on Information and Knowledge Management (CIKM'16). p. 1463–1472. ACM (2016)
3. Andolina, S., Klouche, K., Peltonen, J., Hoque, M., Ruotsalo, T., Cabral, D., Klami, A., Głowacka, D., Floréen, P., Jacucci, G.: IntentStreams: smart parallel search streams for branching exploratory search. In: Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI'15). pp. 300–305. ACM (2015)
4. Auer, P.: Using confidence bounds for exploitation-exploration trade-offs. Journal of Machine Learning Research **3**(Nov), 397–422 (2002)
5. Backhausen, D.T.J.: Adaptive ir for exploratory search support. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'12). p. 992. ACM (2012). https://doi.org/10.1145/2348283.2348416
6. Belkin, N.J., Kantor, P., Fox, E.A., Shaw, J.A.: Combining the evidence of multiple query representations for information retrieval. Information Processing & Management **31**(3), 431 – 448 (1995). https://doi.org/10.1016/0306-4573(94)00057-A
7. Bilenko, M., White, R.W.: Mining the search trails of surfing crowds: identifying relevant websites from user activity. In: Proceedings of the 17th International Conference on World Wide Web (WWW'08). pp. 51–60. ACM (2008)
8. Brooke, J., et al.: Sus-a quick and dirty usability scale. Usability Evaluation in Industry **189**(194), 4–7 (1996)
9. Byström, K., Kumpulainen, S.: Vertical and horizontal relationships amongst task-based information needs. Information Processing & Management **57**(2), 102065 (2020). https://doi.org/10.1016/j.ipm.2019.102065
10. Carmel, D., Yom-Tov, E., Darlow, A., Pelleg, D.: What makes a query difficult? In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06). pp. 390–397. ACM (2006)
11. Chang, J.C., Hahn, N., Kittur, A.: Mesh: Scaffolding comparison tables for online decision making. In: Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST'20). p. 391–405. ACM (2020). https://doi.org/10.1145/3379337.3415865
12. Chang, J.C., Hahn, N., Perer, A., Kittur, A.: SearchLens: Composing and capturing complex user interests for exploratory search. In: Proceedings of the 24th International Conference on Intelligent User Interfaces (IUI'19). p. 498–509. ACM (2019). https://doi.org/10.1145/3301275.3302321
13. Croft, W.B.: Combining approaches to information retrieval. In: Croft, W.B. (ed.) Advances in Information Retrieval: Recent Research from the Center for Intelligent Information Retrieval, pp. 1–36. Springer US (2000). https://doi.org/10.1007/0-306-47019-5_1
14. Du, J.T., Evans, N.: Academic users' information searching on research topics: characteristics of research tasks and search strategies. The Journal of Academic Librarianship **37**(4), 299 – 306 (2011). https://doi.org/10.1016/j.acalib.2011.04.003
15. Fluit, C., Sabou, M., van Harmelen, F.: Ontology-based information visualization: toward semantic web applications. In: Geroimenko, V., Chen, C. (eds.) Visualizing

the Semantic Web: XML-based Internet and Information Visualization. Springer Verlag (2002)
16. Hearst, M.A.: Search User Interfaces. Cambridge University Press, 1st edn. (2009)
17. Huang, J., Lin, T., White, R.W.: No search result left behind: branching behavior with browser tabs. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM'12). pp. 203–212. ACM (2012)
18. Huang, J., White, R.W.: Parallel browsing behavior on the web. In: Proceedings of the 21st ACM Conference on Hypertext and Hypermedia (HT'10). pp. 13–18. ACM (2010)
19. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. ACM Transactions on Information Systems (TOIS) **20**(4), 422–446 (2002)
20. Krishnamurthy, Y., Pham, K., Santos, A., Freire, J.: Interactive exploration for domain discovery on the web. In: Proceedings of the ACM KDD Workshop on Interactive Data Exploration and Analytics (IDEA'16). pp. 64–71 (2016)
21. Marchionini, G.: Exploratory search: from finding to understanding. Communications of the ACM **49**(4), 41–46 (2006)
22. Mizzaro, S., Mothe, J.: Why do you think this query is difficult?: a user study on human query prediction. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'16). pp. 1073–1076. ACM (2016)
23. Moraes, F., Santos, R.L., Ziviani, N.: On effective dynamic search in specialized domains. In: Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR'17). pp. 177–184. ACM (2017)
24. Nitsche, M., Nürnberger, A.: Quest: Querying complex information by direct manipulation. In: Proceedings of the International Conference on Human Interface and the Management of Information (HIMI'13). pp. 240–249. Springer (2013)
25. Oppenlaender, J., Kuosmanen, E., Goncalves, J., Hosio, S.: Search support for exploratory writing. In: Lamas, D., Loizides, F., Nacke, L., Petrie, H., Winckler, M., Zaphiris, P. (eds.) International Conference on Human-Computer Interaction (INTERACT'19). pp. 314–336. Springer International Publishing (2019)
26. Peltonen, J., Belorustceva, K., Ruotsalo, T.: Topic-relevance map: visualization for improving search result comprehension. In: Proceedings of the 22nd International Conference on Intelligent User Interfaces (IUI'17). pp. 611–622. ACM (2017). https://doi.org/10.1145/3025171.3025223
27. Peltonen, J., Strahl, J., Floréen, P.: Negative relevance feedback for exploratory search with visual interactive intent modeling. In: Proceedings of the 22nd International Conference on Intelligent User Interfaces (IUI'17). pp. 149–159. ACM (2017). https://doi.org/10.1145/3025171.3025222
28. Pu, P., Chen, L., Hu, R.: A user-centric evaluation framework for recommender systems. In: Proceedings of the Fifth ACM Conference on Recommender Systems (RecSys'11). pp. 157–164. ACM (2011)
29. Ruotsalo, T., Peltonen, J., Eugster, M., Głowacka, D., Konyushkova, K., Athukorala, K., Kosunen, I., Reijonen, A., Myllymäki, P., Jacucci, G., et al.: Directing exploratory search with interactive intent modeling. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM'13). pp. 1759–1764. ACM (2013)
30. Ruotsalo, T., Peltonen, J., Eugster, M.J., Głowacka, D., Floréen, P., Myllymäki, P., Jacucci, G., Kaski, S.: Interactive intent modeling for exploratory search. ACM Transactions on Information Systems (TOIS) **36**(4), 44 (2018)

31. Russell-Rose, T., Chamberlain, J., Shokraneh, F.: A visual approach to query formulation for systematic search. In: Proceedings of the Conference on Human Information Interaction and Retrieval (CHIIR'19). pp. 379–383. ACM (2019)
32. di Sciascio, C., Sabol, V., Veas, E.E.: Rank as you go: user-driven exploration of search results. In: Proceedings of the 21st International Conference on Intelligent User Interfaces (IUI'16). pp. 118–129. ACM (2016). https://doi.org/10.1145/2856767.2856797
33. di Sciascio, C., Veas, E., Barria-Pineda, J., Culley, C.: Understanding the effects of control and transparency in searching as learning. In: Proceedings of the 25th International Conference on Intelligent User Interfaces (IUI'20). p. 498–509. ACM (2020). https://doi.org/10.1145/3377325.3377524
34. Shneiderman, B.: The eyes have it: a task by data type taxonomy for information visualizations. In: Proceedings 1996 IEEE Symposium on Visual Languages. pp. 336–343 (1996)
35. Shokouhi, M., Si, L.: Federated search. Foundations and Trends in Information Retrieval **5**(1), 1–102 (2011). https://doi.org/10.1561/1500000010
36. Srinivas, N., Krause, A., Kakade, S., Seeger, M.: Gaussian process optimization in the bandit setting: no regret and experimental design. In: Proceedings of the 27th International Conference on Machine Learning (ICML'10). pp. 1015–1022. Omnipress (2010)
37. Umemoto, K., Yamamoto, T., Tanaka, K.: Search support tools. In: Fu, W.T., van Oostendorp, H. (eds.) Understanding and Improving Information Search: A Cognitive Approach. pp. 139–160. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-38825-6_8
38. Venna, J., Peltonen, J., Nybo, K., Aidos, H., Kaski, S.: Information retrieval perspective to nonlinear dimensionality reduction for data visualization. Journal of Machine Learning Research **11**(Feb), 451–490 (2010)
39. Verbert, K., Parra, D., Brusilovsky, P.: Agents vs. users: visual recommendation of research talks with multiple dimension of relevance. ACM Transactions on Interactive Intelligent Systems (TiiS) **6**(2) (Jul 2016). https://doi.org/10.1145/2946794
40. White, R.W., Roth, R.A.: Exploratory Search: Beyond the Query-Response Paradigm. Synthesis Lectures on Information Concepts, Retrieval, and Services, Morgan & Claypool Publishers (2009)
41. Xiang, B., Jiang, D., Pei, J., Sun, X., Chen, E., Li, H.: Context-aware ranking in web search. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 451–458. SIGIR '10, Association for Computing Machinery, New York, NY, USA (2010). https://doi.org/10.1145/1835449.1835525
42. Yogev, S.: Exploratory search interfaces: Blending relevance, diversity, relationships and categories. In: Proceedings of the Companion Publication of the 19th International Conference on Intelligent User Interfaces (IUI Companion '14). p. 61–64. ACM (2014). https://doi.org/10.1145/2559184.2559187
43. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01). pp. 334–342. ACM (2001)