Towards a Game-Theoretic Framework for Text Data Retrieval

ChengXiang Zhai
Department of Computer Science, University of Illinois at Urbana-Champaign
Email: czhai@illinois.edu

Abstract

The task of text data retrieval has traditionally been defined as to rank a collection of text documents in response to a query. While this definition has enabled most research progress so far, it does not model accurately the actual retrieval task in a real search engine application, where users tend to be engaged in an interactive process with multipe queries, and optimizing the overall performance of a search engine system on an entire search session is far more important than its performance on an individual query. This paper presents a new game-theoretic formulation of the text data retrieval problem where the key idea is to model text retrieval as a process of a search engine and a user playing a cooperative game, with a shared goal of satisfying the user's information need (or more generally helping the user complete a task) while minimizing the user's effort and the operation cost of the retrieval system. Such a game-theoretic framework naturally optimizes the overall utility of an interactive retrieval system over a whole search session, thus breaking the limitation of the traditional formulation that optimizes ranking of documents for a single query and can optimize the collaboration of a search engine and its users, thus maximizing the "combined intelligence" of a system and users. Although the framework was motivated by text data retrieval, it is actually quite general and potentially applicable to all kinds of information service systems.

1 Introduction

Text data are those data that are encoded in human natural languages such as English and Chinese. They are unique in that they are generated by humans and also meant to be consumed by humans. As a result, they play a very important role in our life. For example, most human knowledge is encoded in the form of text data; scientific knowledge almost exclusively exists in scientific literature. Because natural languages are the tools of communication by humans, we can usually describe other media such as video or images using text data, facilitating understanding of videos and images. From data mining perspective, because of the rich semantic content in text data, we can expect to discovery all kinds of knowledge from text data, especially knowledge about people's preferences and opinions, which are often best reflected in the text data produced by them and may be hard to obtain from other kinds of machine-generated data.

The amount of text has recently grown dramatically due to the digitalization of information and the widespread deployment of tools for people to easily produce and consume text online. This creates a significant challenge for humans to consume and make use of all the text data in a timely manner Logically, in order to make use of

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

"big text data," we would first need to select the most relevant text data from a large collection of text documents and filter out many non-relevant text data; this not only helps filter out noise, but also significantly reduces the size of the data that we actually have to digest [26]. The most useful tool for supporting this first step in text data applications is a text data retrieval system, or a search engine. Indeed, Web search engines are now essential tools in our daily life, and they help us find relevant documents from the Web quickly, effectively addressing the problem of information overload. There are also many other examples of search engine applications such as product search, social media search, and scientific literature search. In general, wherever we have text data, we would need a search engine.

Since the accuracy of a search engine directly affects our productivity, it is very important to study how to improve the accuracy of all search engines so that we can save people's time. It is especially beneficial to develop general techniques that can improve *all* search engines since such techniques can be immediately used in all kinds of search engine applications to generate benefit in all application domains. For this reason, the optimization of general text retrieval models has been a long-standing important fundamental reseach question in the information retrieval community (see, e.g., [23, 16, 20]), and many general informtion retrieval models have been proposed and tested (see [25] for a detailed review of all these models).

So far, the task of text data retrieval has been mostly defined as to generate an optimal ranking of documents in response to a user's query. The ranking is usually based on a scoring function defined on a query-document pair to generate a relevance score. The theoretical justification for such a ranking formulation is the Probability Ranking Principle (PRP) [15], which states that returning a ranked list of documents in descending order of probability that a document is relevant to the query is the optimal strategy under two assumptions: (1) The utility of a document (to a user) is independent of the utility of any other document. (2) A user would browse the results sequentially. The intuition captured by PRP is the following: if a user sequentially examines one doc at each time, wed like the user to see the very best ones first, which makes much sense.

While the PRP has enabled most research progress in text retrieval so far, it does not model accurately the actual retrieval task in a real text retrieval application since neither assumption actually holds in practice; even the extended PRP for interactive retrieval [8] is still limited due to the assumption of sequential browsing. Moreover, retrieval is in general an interactive process often with multipe queries formulated, and thus optimizing the overall performance of a text retrieval system on an entire search session is far more important than its performance on an individual query. Unfortunately, the current formulation of the retrieval task makes it impossible to optimize over an entire session.

To address these limitations, we present a novel game-theoretic formulation of the text data retrieval problem where the key idea is to model text retrieval as a process of a search engine and a user playing a cooperative game, with a shared goal of satisfying the user's information need (or more generally helping the user complete a task) while minimizing the user's overall effort and the operation cost of the retrieval system. Such a game-theoretic framework offers two important benefits. First, it naturally suggests optimization of the overall utility of an interactive retrieval system over a whole search session, thus breaking the limitation of the traditional formulation of optimizing the ranking of documents for a single query. Second, it models the interactions between users and a search engine, and thus can optimize the collaboration of a search engine and its users, maximizing the "combined intelligence" of a system and users [4].

2 Text Retrieval as Cooperative Game Playing: Basic Idea

How can we optimize all search engines in a general way? This question is not well defined until we clearly define what a search engine is, and what is an optimal search engine. The previous section has made it clear that defining a retrieval task as optimizing a ranked list in response to a query has many limitations. However, what is the most general way to define the task of text retrieval that would address all the limitations then? We argue that the most general way to define a retrieval task is to define it as a search engine system playing a sequential cooperative game with the user with a shared goal of helping the user finish the information seeking task with minimum overall user effort and minimum operation cost of the system.

In the simplest setting, we will consider just one user, though the idea can be easily generalized to include multiple users. In such a case, the game has two players: player 1 is the search engine, while player 2 is the user. The rules of the game are as follows:

1. Players take turns to make "moves" (just as in a board game like chess). A move is an action taken by user/system.

2. The first move is usually made by the user in the case of search, i.e., the entering of a query by the user (though in a recommender system, the system can also make the first move).

3. For each move of the user, the system makes a response move, i.e., shows an interactive interface; for each move of the system (i.e., each interactive interface), the user makes a response move (i.e., takes an action on the interface).

4. The game is over when the user has finished the information seeking task or decided to abandon the search (failed to find the needed information).

The objective of the game is to help the user complete the (information seeking) task with minimum overall effort and minimum operating cost for the search engine.

Such a game-theoretic view of the retrieval problem can be easily understood by analyzing the interaction process of the current search engine with a user as shown in Figure 1. In general, we may assume that the user has the goal of finding useful information with minimum effort, while the system has the goal of helping user find useful information with minimum effort as well as minimum system cost. The user's query is the very first move of the game, made by the user (denoted as Action A_1). In response to this move, the system now has to make a decision over what action/move to take. The decision can be "which information items to present" and "how to present these items" in response to the user's query. Now suppose the system has decided to show a certain search result denoted by R_1 in an interface. The user would look at the interface and decide his/her next move. That is, the user would decide which items to view. The user then chooses an item to view (action A_2). In response to this move, the system would then further decide which move to make in response to the new action taken by the user. The decision questions here can include which aspects/parts of the selected information item

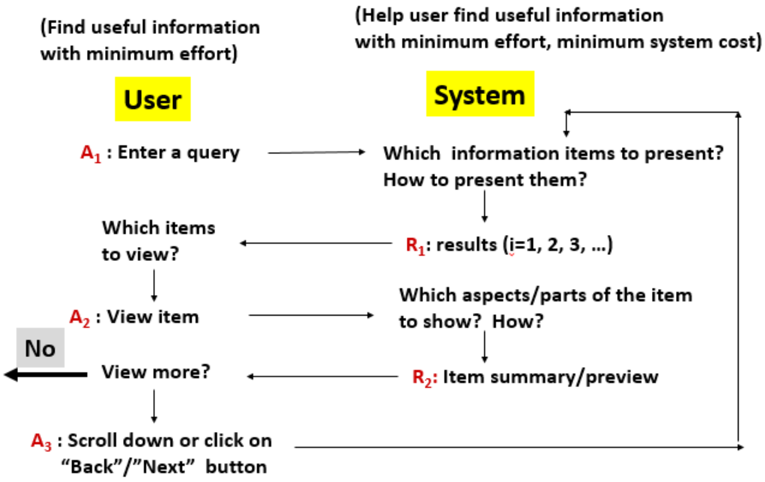
(Find useful information
with minimum effort)
User
(Help user find useful information
with minimum effort, minimum system cost)
System
A1 : Enter a query
Which information items to present?
How to present them?
Which items
to view?
R1: results (i=1, 2, 3, …)
A2 : View item
Which aspects/parts of the item
to show? How?
No
View more?
R2: Item summary/preview
A3 : Scroll down or click on
"Back"/"Next" button

Figure 1: Retrieval process as a sequential game.

by the user should be shown to the user, and how to show them? In a regular retrieval system, such decisions are often ignored as the engine often just returns the document to the user, and the user would browse the document. However, if we are to optimize the response from the engine, we could imgine to display only the most relevant part of a long document with possibility to navigate into other parts, or show a summary that can further support navigation into the original content. Furthermore, if the system shows only the most relevant part of a document, the system can also adaptively optimize the presentation of the remaining part if the user chooses to scroll up or down (e.g., if the user has spent a lot of time reading the displayed relevant part, the system can infer the displayed content is interesting to the user, whereas if the use didn't really spend time to read it and directly chose to scroll up/down, the system would know to avoid displaying similar content). When viewed as a game, such opportunities for optimization would be obvious. After the user finishes examining a result, the user would again face a decision: should the user view more? If the user decides to view more, the user may decide to click on a "Next" button, which we again would view as a new move by the user. From a system's perspective, it could do exactly the same as what it did when it responded to the original qeury, i.e., the system would again decide which items to show and how to show them. This means after the user clicks on the "Next" button, the results shown to the user can be very different from the original next page since by this time, the system would have known much more about the user's need (based on analyzing the user's behavior on the first page). The process can go on and on until the user decides that he/she has viewed sufficient items and decides to stop.

There are many benefits of formulating the retrieval problem (i.e., the task of a search engine) as playing such a cooperative game (and thus also defining an optimal search engine as one that plays such a game with an optimal strategy). First, this gives us a formal framework to naturally integrate research in user studies, evaluation, retrieval models, and efficient implementation of retrieval systems since the optimization of the objective of such a game involves research in all these areas. Second, it provides a unified roadmap for identifying unexplored important retrieval research topics. Third, it emphasizes on optimization of an entire session instead of that on a single query (i.e., optimizing the chance of winning the entire game). Finally, it enables optimization of the collaboration of machines and users, i.e., maximizing collective intelligence [4].

The new formulation also raises many interesting new questions which we will discuss later in this paper. First, how should we design a text retrieval game? Specifically, how should we design moves for the user and the system, how should we design the objective of the game, and how can we go beyond search to support access and task completion? Second, how can we formally define the optimization problem and compute the optimal strategy for the IR system? Specifically, how do we characterize a text retrieval game, which category of games does a text retrieval game fit, to what extent can we directly apply existing game theory, and what new challenges must be solved? Finally, how should we evaluate such a system? In the rest of the paper, we will briefly address some of these questions with a focus on presenting a formal framework for optimizing the retrieval decisions in such a game.

3 Text Retrieval as Cooperative Game Playing: Formal Framework

It is important to point out that the game of retrieval is not a zero-sum game, thus it is different from a game such as chess in this sense. However, it shares similarity with chess in that they both involve optimization of sequential decisions over a horizon of multiple moves. Indeed, just as in chess where it sometimes makes sense to sacrifice a piece in order to win a game, it also makes sense for a search engine to sometimes take a "local loss" in order to gain more in the overall session. An example is when the query is ambiguous, a search engine can ask the user to clarify whether the word "Jaguar" in the query means a car or an animal. This move is not optimal because it is not as useful to the user as if the system simply takes a guess of the sense of "jaguar" and provides some search results, but it helps to optimize the results in all future moves once the system knows the intended sense of "jaguar."

The challenge is, however, how do we mathematically optimize such sequential decisions? How do we know when the search engine should ask a question and when it shouldn't? To address these questions, we must first formalize our decision problem by introducing notations to denote all the important variables. This is illustrated in Figure 2.

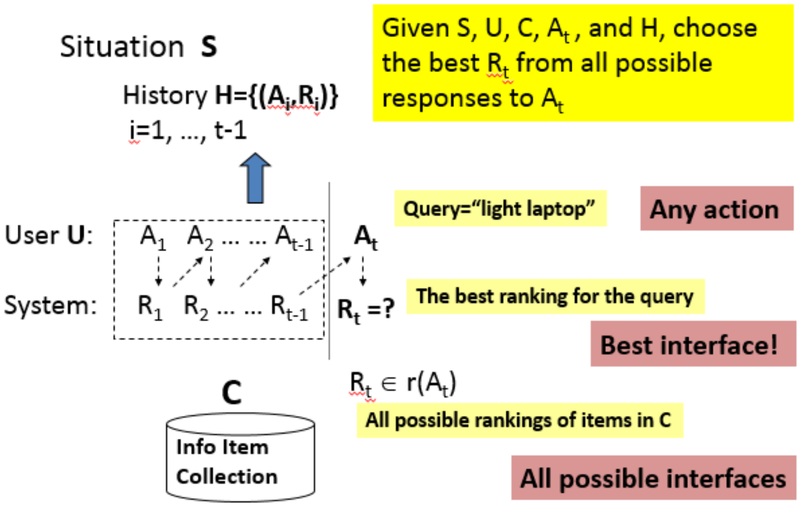
Situation S
History H={(A_i,R_i)}
i=1, ..., t-1
Given S, U, C, A_t, and H, choose the best R_t from all possible responses to A_t
User U:
System:
A_1 A_2 ... ... A_{t-1}
R_1 R_2 ... ... R_{t-1}
A_t
R_t =?
Query="light laptop"
Any action
The best ranking for the query
Best interface!
C
Info Item Collection
$R_t \in r(A_t)$
All possible rankings of items in C
All possible interfaces

Figure 2: Formalization of the retrieval decision in a retrieval game.

We use A_i to denote an action taken by the user U, and R_i the corresponding response (system action) by the system. At time t, the system needs to choose a response R_t in response to the current user action A_t. For convenience, we would use $H = \{(A_i, R_i)\}$ to denote all the history information we can observe about the user-system interactions. We denote the collection of information items to be searched by C and further use a generic variable S to denote the current search scenario which can also be interpreted as representing the general context of the search (e.g., time and location of the search), which may also be a factor affecting the system's decision. For example, there may be some general preferences in the Holiday shopping season in a year or when an important international event has just happened.

With these variables, the retrieval decision problem can now be framed as: Given situation S, user U, collection C, history H, and the current action A_t, choose the best response R_t from all the possible responses to action A_t, which we denote by $r(A_t)$.

Note that such a formulation is much more general than the current formulation of the retrieval problem (which is to optimize ranking of documents for a query) since a user's action can be, theoretically speaking, any action that a user can take, including not just entering a query, viewing a document, or clicking on the "Next" button, but also any keystroke, any cursor movement, and many others. Also, the system's response to a user action can also easily go beyond the normal responses such as showing a ranked list of documents to potentially include *any* interaction interface that the system can show to the user, which further opens up many possibilities of actions that a user can take for further interaction with the system. Naturally, the traditional formulation of the retrieval problem can be easily seen as a special case where the user's current action A_t would be the query entered by the user, the response of the system R_t is a ranked list of documents in collection C, and the set $r(A_t)$ consists of all possible rankings of these documents.

With the introduced notations, we can now use Bayesian decision theory to formally frame the problem as a statistical decision problem, which we illustrate in Figure 3. The formulation can be regarded as a generalization of the risk minimization framework for retrieval [24]. In this figure, we see that our observed variables include S, U, H, C, and A_t, and the space of actions to choose is given by $r(A_t) = \{r_1, ..., r_n\}$, where r_i is a potential action that the system can take. In order to assess which action is a good action to take, we introduce a loss function $L(r_i, M, S)$ which depends on the candidate action r_i, the situation S, and also a new variable M, which is a user model variable that encodes "everything" that we need to know about the user at the point for deciding which action to take. Intuitively, the system should choose a response that would minimize the value of the loss function. That is, we want to choose an r_i that has the smallest $L(r_i, M, S)$ (as compared with other possible responses).

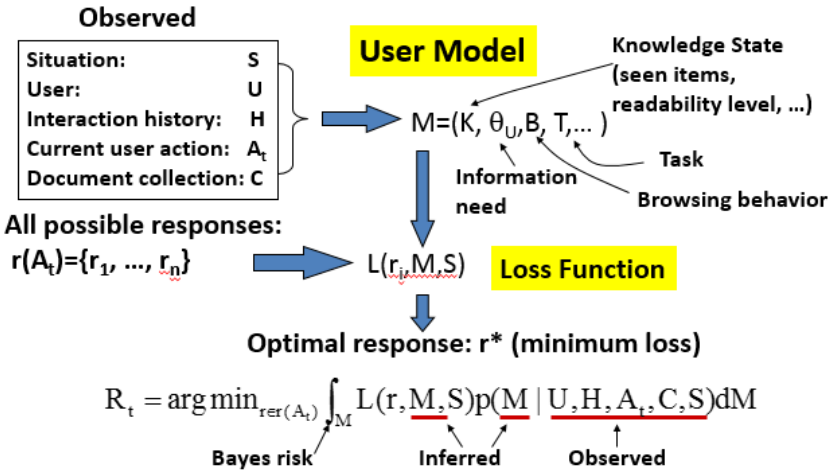

Figure 3: Bayesian decision theory applied to optimization of retrieval decisions.

However, we have note yet said what is exactly the user model variable M. Theoretically speaking, M can potentially encode all the detailed information that we might know directly or indirectly about the user. As a minimum, though, M would contain information about the user's information need, which initially can only be inferred based on the user's query, but can be updated if we know more information about the user. This was denoted by θ_U in the figure. M may also contain information about the user's knowledge status (which is denoted by K and can contain the items already viewed by the user, i.e., "known information" to the user), the user's reading level (which can be inferred based on the documents that the user has read/skipped), a user's browsing behavior, and any information about the user's task.

If M is clearly specified, our decision problem would be relatively easy as we just need to compute $L(r_i, M, S)$ for every r_i and choose the one minimizing our loss function. However, in general, we cannot clearly specify M, and thus the best we can do is to infer M based on all the observed variables using the posterior distribution $p(M|U, H, A_t, C, S)$, which captures our belief about M after we observe all the observables. Considering this uncertainty, the solution to our problem can be defined as the response that minimizes the expected loss (also called expected risk, or Bayes risk) as shown in the figure.

The computation of the Bayes risk involves the computation of an integral over the space of all possible user models, which is clearly intractable. It is thus interesting to look at an approximation of this integral by considering the mode (highest value) of the posterior distribution of the user model M, which we denote by M^*. Since $M^* = \arg\max_M p(M|U, H, A_t, C, S)$, with this approximation, we have

$$\begin{aligned} R_t &\approx \arg\min_{r\in r(A_t)} L(r, M^*, S)p(M^*|U, H, A_t, C, S) && (8) \\ &= \arg\min_{r\in r(A_t)} L(r, M^*, S) && (9) \end{aligned}$$

This suggests the following two-step procedure for computing the optimal response: 1) Compute an updated user model M^* based on all the currently available information, i.e., compute $M^* = \arg\max_M p(M|U, H, A_t, C, S)$.

2) Given M^*, choose an optimal response to minimize the loss, i.e., compute $R_t = \arg\min_{r \in r(A_t)} L(r, M^*, S)$. We illustrate the game-playing process when the system is taking such a simplified strategy in Figure 4 where the system maintains a user model variable M as an internal state and dynamically updates it in each iteration and then chooses an optimal response to minimize the loss function.

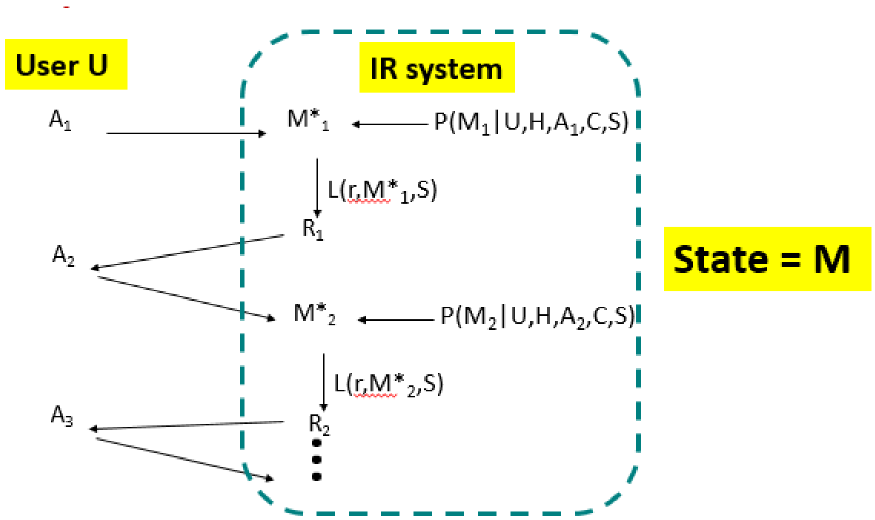

Figure 4: Optimal interactive retrieval based on dynamic updating of user model.

This process suggests that the sequential decision problem here can be naturally modeled with Partially Observable Markov Decision Process (POMDP) by treating M as the state and the update of M as transitions between the states. Indeed, Markov Decision Process (MDP), particularly POMDP, has been recently explored for information retrieval with encouraging results (see e.g., [12, 13]), but the modeling of M in the existing work is still quite limited. When POMDP is used to model interactive retrieval, we would be able to naturally use Reinforcement Learning to enable a retrieval system to learn an optimal decision strategy from its interactions with many users. The idea of search engine as a learning agent was also suggested in the work [9]. We anticipate to see more progress in applying POMDP and reinforcement learning to optimize interactive retrieval in the near future.

An especially interesting direction may be to introduce more detailed user behavior modeling into a retrieval decision framework so that M can be enriched with specific models capturing a user behavior when interacting with the retrieval system. From game-theoretic perspective, it is essential to model the user's decision process. In this direction, the recent work on economics in interactive IR (e.g., [1]) is very promising and can shed light on how we can refine M to model user behavior and further connect the behavior with the loss function to influence the choice of system response.

As often happens in sequential decision making, there is a tradeoff between exploration and exploitation. For example, it may be beneficial to diversify the retrieval results initially in hope of learning more completely about a user's information need, but over-diversification would incur cost because it may decrease the relevance of the current results (as we also include results that may be only marginally relevant). Recent work has explored this problem by using multi-armed Bandit for optimizing online learning to rank (e.g., [10]) and content display and aggregation (e.g., [7]). The game-theoretic framework can capture the exploration and exploitation tradeoff by defining the loss function with consideration of the expected future actions taken by the user.

The evaluation of a system designed based on the game-theoretic framework is challenging. While it is always possible to use A/B test in a real operational system to evaluate different decision strategies of the system based on user responses, it is unclear how we can create a reusable test collection that can be repeatedly used to compare different game-playing algorithms (especially new algorithms yet to be developed) in a controlled manner. One possibility is to create many simulators of different kinds of users based on assumed user behavior and information needs, which can then be combined with a standard retrieval test collection to enable simulation-based evaluation.

4 Instantion of the Game-Theoretic Retrieval Framework

The framework presented in the previous section is very general, but it is mostly a conceptual framework. Thus in order to derive specific models and algorithms that can be implemented in a search engine, we must instantiate all the components in the framework, including (1) actions (i.e., "moves") that can be taken by a user and a system (A_i and $r(A_i)$), (2) user model M, (3) conditional distribution of M given all the observables, $p(M|U, H, A_t, C, S)$, and (4) the loss function $L(r, M, S)$. Clearly, there are many different ways to define these variables; each would lead to a potentially different specific strategy for a retrieval system to use for playing such a cooperative game (i.e., a strategy for optimizing the sequential retrieval decisions in a whole interaction session with a user). A thorough discussion of all these possibilities is beyond the scope of this paper, but we will briefly discuss some possibilities, mostly to suggest even more possibilities are possible.

4.1 Instantiation of "moves" in the game

We first look at the question how to define the moves in such a cooperative game, i.e., the actions that the user and the system can take respectively. It is easy to see that when viewed with this new framework, a current search engine can only play a very simple game with its users with extremely limited moves. Indeed, a user's actions are mostly restricted to entering a query, clicking on a result, scrolling up and down on a page, and clicking other buttons to navigate into additional results. The system's response is also mostly restricted to either produce a ranked list or passively show the requested results in the ranked list already pre-computed by the search engine based on the user's query. This is indeed a very simple game with very limited moves! The new framework suggests that the game can be much more sophisticated with many more possible moves. It is especially useful to include those moves that do not require much effort from the user, but woud help clarify the user's information need since this would help optimize human-computer collaboration by more actively engaging the user in a "dialogue" where the user and system help each other to help the user finish the information seeking task with minimum user effort and minimum system operation cost.

To see some specific moves that can be added to such a game, let us first consider the restrictive set of user actions the current search engine supports. A careful analysis would reveal that even for these actions, the system could have had many more "moves" to make than what is supported by the current search engine. For example, when a user requests the next page, the results on the next page can be completely reranked based on what the system has learned from the user's actions on the current page as done in some systems such as UCAIR [17] and SurfCanyon (http://www.surfcanyon.com/). The system can even go further to summarize the next page by using what the user has seen on the first page as the context. Similarly, when the user scrolls down on a page, the system would also have an opportunity to adaptively customize how to present the rest of the page; for example, the system can have likely non-relevant parts collapsed to save screen space or "decorate" the page with additional navigation buttons as appropriate, which would enable the user to make more moves (each navigation can be viewed as one possible move by the user). Of course, when we offer a user more actions to take, it would further provide more opportunities for the system to respond in different ways.

The framework also enables us to model user actions at different levels and thus allow a system to play the game at different levels. For example, the low level actions of users can include key strokes, mouse clicking and movement, or even eye tracking. When viewing a user's action at this level, we could identify opportunities for responding to a key stroke when the user enters a query with suggestions for query autocompletion. In this sense, query autocompletion or query spelling correction is a natural strategy that can be derived from the game-theoretic framework of text retrieval. If we view query input as a medium-level user action, then we may also view an entire query session with multiple queries entered by the user (e.g., searching for hotels) as one "high-level" action by the user, enabling the system to make a "high-level" response by recommending information relevant to the user's task behind the search session (e.g., the task can be travel), such as recommeding attractions in the city of the hotel. Such a flexibility in modeling user-system interactions makes the framework very general and allows us to selectively model those actions that are most useful for a particular application scenario.

Another interesting example of a response to a user's action of entering query is to ask the user a clarification question such as "did you mean jaguar as a car or animal?" When looking at the optimization at the level of an individual query, such a response is clearly non-optimal as it does not provide any useful information to the user and is thus worse than presenting a ranked list of pages using any standard retrieval function. However, when viewed globally from the perspective of optimizing the entire session, it is possible that the local loss can easily be balanced by the gain in the future interactions due to the clear understanding of the intended sense of "jaguar." In some sense, this is similar to sacrificing a piece in chess to win the game where the loss of a piece would incur an immediate (short-term) loss, but it helps win the game (gains in the long run), and the game-theoretic view of the retrieval problem would allow us to use an algorithm to decide when the system may consider asking such a clarification question (e.g., when the current results are poor and the user has already reformulated the query multiple times).

How to improve search accuracy for long-tail queries is one of the most important and challenging questions faced today in optimizing a search engine. This is because the system does not see many instances of a query, making the machine learning approach ineffective. (In contrast, the popular queries benefit significantly from the many clickthroughs from the users that the system can collect.) To address this problem, we may introduce new moves to allow a user to provide "explanatory feedback" when the search results are poor and the user can't easily reformulate the query either. An example of such a move can be to allow a user to click on a button to select from a menu with choices such as 1) I want documents similar to this one except for not matching "X" where the user would type in "X," or 2) I want documents similar to this one, but also further matching "Y" where the user would type in "Y." Such a move is very natural for a user and does not require much effort from the user, but can be very effective for helping the system accurately learn the user's information need.

4.2 Instantiation of user model M

As already discussed, formally modeling a user's state through the variable M is a very important component in the proposed framework, and a major novelty of the framework over existing ways to model the retrieval process. In general, the formal user model M is intended to capture all the essential knowledge about a user's status for the purpose of optimizing the system's response especially for personalization [21]. We can thus imagine many components that can be included in M. The most important component is the model of the current information need of the user. This component is usually implemented as a term vector [18] or a word distribution in the language model [14] in a current search engine, allowing for matching with a document vector or document word distribution to generate a score of relevance. However, more sophisticated representation is clearly possible, including, e.g., tracking multiple subtopics of interest to the user or negative models for information not interesting to the user.

Another component that is very easy to maintain is a model of the information items that have already been seen or viewed by the user. This component is important for assessing the novelty of any information item that may be displayed to the user in the future. One can also model the reading level of a user, which can be important if the user is an elementary school student since it helps the system avoid presenting topically relevant materials that are beyond the reading level of the user.

The user model M can further include information about the user's browsing behavior. For example, does the user tend to view many results on a page or even go to the next page, or the user often just views the top three to five results? Having this knowledge clearly helps optimizing the design of the interface shown to the user when responding to the user's query. Of course, M can also include many other aspects about the user such as the user's task (which can often be inferred based on the multiple queries entered by the user; for example finding flight tickets and booking hotel can suggest a task of travel). The model can even include an estimate of the patience level of the user, which can also affect the optimization of the system's response.

It is easy to see that M can potentially include all kinds of findings about user preferences and behaviors from user studies that may be relevant to optimizing a search task. This user model thus provides a formal and principled way to integrate relevant findings by human-computer interaction researchers into a search engine to optimize its utility. The user modeling component of the framework can also be regarded as a way to formalize some existing theories about user modeling in information retrieval, notably the Anomalous State of Knowledge (ASK) theory [3] and the Cognitive Information Retrieval Theory [11].

4.3 Inference of user model

The inference of user model M is based on the posterior distribution $p(M|U, H, A_t, C, S)$, which enables inference of M based on everything the system has available so far about the user and his/her interactions. The instantiation of this distribution can be based on findings from user studies or machine learning using user interaction log data for training. The current search engines mostly focused on estimating and updating the information need model, which is only part of the general user model M. Future search engines must also infer and update many other variables about the user such as the inference about the user's task, exploratory vs. fixed item search, reading level, and browsing behavior. Some existing work can be leveraged for making such inferences (e.g., reading level [5], modeling decision point [22]).

4.4 Instantiation of loss function

In general, the loss function $L(r, M, S)$ should combine measures of 1) *Utility* of response r for a user modeled as M to finish the user's task in situation S; 2) *Effort* of a user modeled as M in situation S; and 3) *Cost* of system performing r. The tradeoff between these three aspects would inevitably vary across users and situations. The utility of response r can be defined as a sum of the *immediate* utility of r to the user and the *future* utility derived from future interactions of the user enabled by response r, which depends on the user's interaction behavior.

Formalization of the utility function requires research on evaluation, task modeling, and user behavior modeling. The traditional evaluation measures, such as Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG), tend to use very simple user behavior model (i.e., sequential browsing) and use a straightforward combination of effort and utility. They would need to be extended to incorporate more sophisticated user behavior models (e.g., in the line of [6, 19, 2]) to more accurately model the utility and user effort and enable more flexible tradeoff.

The system operation cost can be modeled based on the expected consumption of computing resources such as CPU time, memory, and disk space by computing the response r. In a more general sense, the cost may even include the cost on hiring humans to help answer a query (crowdsourcing), which may then justify a response r that is based on human computation, rather than a retrieval algorithm.

4.5 The Interface card model

The interface card model [27] is a general instantiation of the game-theoretic framework where a system response is defined very generally as any interaction interface, and the retrieval decision is thus reduced to the decision on which interface to choose. The objective function to be optimized reflects the expected surplus of presenting a particular interaction interface which is calculated based on the reward (gain) of relevant information obtained by the user adjusted by the cost due to the effort that the user has to make in order to gain the relevant information. The expectation is taken with respect to a distribution of all possible actions that a user can take. With further assumptions, the model is shown to be able to generalize PRP by relaxing both assumptions made in PRP (i.e., sequential browsing and independent utility of documents). Moreover, the model is shown to be able to automatically determine the optimal layout of a navigation interface in adaptation to the system's confidence in inference about the user's information need and the screen size of the display. This is a very promising model because it opens up a new direction of computationally optimizing the layout of an interface, which can have broad impact beyond a retrieval system and may be especially useful for optimizing the interface design for mobile phone users where the screen size is very small, thus requiring efficient use of the limited space.

5 Conclusions

In this paper, we addressed some fundamental questions about how to optimize text data retrieval systems in a general way and proposed to define the retrieval task as having the system to play a cooperative retrieval game with the user with a shared objective to complete the user's task with minimum overall user effort and minimum system operation cost. We used Bayesian decision theory to formally frame the problem as a statistical decision problem with a formal model of users as a key component for optimizing retrieval decisions. The game-theoretic framework can potentially integrate research in user studies, evaluation, retrieval models, and efficient implementation of text retrieval systems in a single unified principled framework and also serves as a roadmap for identifying unexplored important IR research topics. The two important benefits of the framework are 1) natural optimization of the utility over an entire session instead of that on a single query and 2) optimization of the collaboration of machines and users (thus maximizing collective intelligence). We also briefly discussed how to instantiate all the major components of the framework so as to derive more specific models that can be implemented in a search engine and introduced the Interface Card Model as a specific example which can optimize the interface design for navigation into relevant items. We show that the game-theoretic framework can model user actions in a very general way and at different levels of granularity.

To fully implement a system based on such a new framework, there are obviously many challenges to solve, and we would need to integrate research in multiple areas including 1) formal modeling of users and tasks, 2) modeling and measuring system cost, 3) machine learning, particularly reinforcement learning to enable the system to effectively update user model, 4) efficient algorithms to enable fast response by the system, and 5) evaluation methodology for evaluating such an interactive system. While the framework was proposed for the problem of text data retrieval where involvement of users in the loop is especially important, we believe that the general idea of the framework and many components may also be broadly applicable to other interactive systems where optimization of human-machine collaboration is important.

References

[1] L. Azzopardi. Modelling interaction with economic models of search. In *Proceedings of ACM SIGIR 2014*, pages 3–12, New York, NY, USA, 2014. ACM.

[2] F. Baskaya, H. Keskustalo, and K. Järvelin. Modeling behavioral factors ininteractive information retrieval. In *Proceedings of ACM CIKM 2013*, CIKM '13, pages 2297–2302, New York, NY, USA, 2013. ACM.

[3] N. J. Belkin. Anomalous states of knowledge as a basis for information retrieval. *The Canadian Journal of Information Science*, (5):133–143, 1980.

[4] N. J. Belkin. Intelligent information retrieval: Whose intelligence? In *Proceedings of the Fifth International Symposium for Information Science*, 1996.

[5] K. Collins-Thompson, P. N. Bennett, R. W. White, S. de la Chica, and D. Sontag. Personalizing web search results by reading level. In *Proceedings of ACM CIKM 2011*, pages 403–412, 2011.

[6] A. P. de Vries, G. Kazai, and M. Lalmas. Tolerance to irrelevance: A user-effort evaluation of retrieval systems without predefined retrieval unit. In *Coupling Approaches, Coupling Media and Coupling Languages for Information Retrieval - RIAO 2004*, pages 463–473, 2004.

[7] F. Diaz. Integration of news content into web results. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 182–191, New York, NY, USA, 2009. ACM.

[8] N. Fuhr. A probability ranking principle for interactive information retrieval. *Inf. Retr.*, 11(3):251–265, June 2008.

[9] K. Hofmann. Fast and reliable online learning to rank for information retrieval, 2013. Doctoral Dissertation.

[10] K. Hofmann, S. Whiteson, and M. de Rijke. Balancing exploration and exploitation in learning to rank online. In *Proceedings of ECIR 2011*, pages 251–263, 2011.

[11] P. Ingwersen. Cognitive perspectives of information retrieval interaction: elements of a cognitive ir theory. *Journal of Documentation*, 52:3–50, 1996.

[12] X. Jin, M. Sloan, and J. Wang. Interactive exploratory search for multi page search results. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13, pages 655–666, New York, NY, USA, 2013. ACM.

[13] J. Luo, S. Zhang, and H. Yang. Win-win search: Dual-agent stochastic game in session search. In *Proceedings of ACM SIGIR 2014*, pages 587–596, New York, NY, USA, 2014. ACM.

[14] J. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the ACM SIGIR'98*, pages 275–281, 1998.

[15] S. E. Robertson. The probability ranking principle in ı°. *Journal of Documentation*, 33(4):294–304, Dec. 1977.

[16] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, 1983

[17] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In *Proceedings of CIKM 2005*, 2005.

[18] A. Singhal and C. Buckley and M. Mitra. Pivoted document length normalization, In *Proceedings of ACM SIGIR 1996*, pages 21–29, 1996.

[19] M. D. Smucker and C. L. Clarke. Time-based calibration of effectiveness measures. In *Proceedings of ACM SIGIR 2012*, pages 95–104, New York, NY, USA, 2012. ACM.

[20] K. Sparck Jones and P. Willett, editors. *Readings in Information Retrieval.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[21] J. Teevan, S. T. Dumais, and E. Horvitz. Potential for personalization. *ACM Trans. Comput.-Hum. Interact.*, 17(1):4:1–4:31, Apr. 2010.

[22] P. Thomas, A. Moffat, P. Bailey, and F. Scholer. Modeling decision points in user search behavior. In *Proceedings of the 5th Information Interaction in Context Symposium*, IIiX '14, pages 239–242, New York, NY, USA, 2014. ACM.

[23] C. J. van Rijsbergen. *Information Retrieval*, Butterworths, London, 1979.

[24] C. Zhai and J. Lafferty. A risk minimization framework for information retrieval. *Information Processing and Management*, 42(1), pages 31-55, 2006.

[25] C. Zhai. *Statistical Language Models for Information Retrieval.* Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2008.

[26] C. Zhai and S. Massung. *Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining.* ACM and Morgan & Claypool Publishers, 2016.

[27] Y. Zhang and C. Zhai. Information retrieval as card playing: A formal model for optimizing interactive retrieval interface. In *Proceedings of ACM SIGIR 2015*, pages 685–694, New York, NY, USA, 2015. ACM.