Is the First Query the Most Important: An Evaluation of Query Aggregation Schemes in Web Search Sessions

No Institute Given

Abstract. Web users often issue a series of related queries, which form a search session, to fulfill complicated information needs. Query aggregation utilizes and combines multiple queries for session search. Prior research has demonstrated that query aggregation is an effective technique for session search that can generally achieve a 15% $\sim$ 20% nDCG@10 improvement on finding relevant documents for the last query. When aggregating queries in a session, how to put weights on them for the most effective search is an interesting question. Is the first query or the last query plays a more important role and thus deserves a higher weight in the session? This paper evaluates various query aggregation schemes and proposes a new three-step query aggregation method. Evaluation on TREC 2011 and TREC 2012 Session Tracks shows that the proposed scheme works very well and significantly outperforms the best TREC systems.

1 Introduction

Complicated search tasks often require more than one queries. These queries form a search session and work as a whole to fulfill an information need. Session search [7, 3, 6, 11, 13, 18, 23] is the Information Retrieval (IR) task that retrieves documents for a session.

Session search happens frequently on the social media and the Web, such as planning a trip, consulting for medical choices, or looking for a good kindergarten. In a typical session, a user starts from an initial query and then goes through the returned search results and chooses some documents to examine. The user may find that the search results satisfy the information need or that the returned documents are insufficient. This motivates the user to modify the query and examine the newly returned documents. This process continues until user's information need is fulfilled.

The TREC (Text REtrieval Conference) 2010-2012 Session tracks [17, 15, 16] have strongly promoted research in session search in academia. The TREC sessions are composed of sequences of queries $q_1, q_2, \cdots, q_{n-1}$, and q_n. TREC Session Tracks assume that the last query is most likely satisfies the information need and thus regards it as most important. Therefore, the task is to retrieve relevant documents for the last query q_n. Table 1 shows examples from the TREC 2012 Session track [16].

The user interactions of TREC Session data were created by NIST assessors. The assessors were given topics as information need and interacted with the Yahoo! BOSS search engine[1] for each topic. The interactions were collected as sessions. A session contained a series of queries,where each except the last one was associated with top k pages returned by Yahoo! BOSS search engine and which the user had clicked. The topics covered large scope in content like daily life, technology, and literal arts. Moreover, the queries were created by assessors who were not necessarily specialists of the given topic. Therefore, these sessions reflect real search behaviors.

Although TREC Session task focuses on retrieval for the last query, TREC participants [1, 6, 11] have demonstrated that using all queries in a session is an effective technique for session search. Particularly, using all queries can generally achieve a 15% $\sim$ 20% nDCG@10 improvement over only using the last query. We term the technique of utilizing and combining multiple queries in a session *"query aggregation"*.

Figure 1 shows the search accuracy (measured by nDCG@10 [5]) improvement of runs submitted to the TREC 2012 Session track [16]. In all 27 submitted runs, 21 applied query aggregation of which 17 observed improvements of search accuracy over not using query aggregation. The average nDCG@10 achieved a significant absolute gain of 18%.

In this paper, we investigate this promising technique, *query aggregation*, and its impact on session search. The most frequently used query aggregation scheme for session search is uniform aggregation which treats every query in a session equally important. Several TREC 2011 and TREC 2012 systems have adopted this scheme. For instance, [1] expands each previous query's anchor text in search results, then reformulates the last query by concatenating the expanded previous queries with equal weights. [6] weighs individual queries equally when applying language modeling for document retrieval.

[1] http://developer.yahoo.com/boss/search/.

Table 1. Examples of TREC 2012 Session queries.

session 7	session 47
q_1.pocono resort	q_1.pseudocyesis
q_2.skytop lodge directions	q_2.pseudocyesis epidemiology
q_3.Philidelphia car rental	q_3.pseudocyesis history
session 14	session 13
q_1.Kursk submarine 2000	q_1.kursk russian sinking
q_2.Kursk submarine 2000 politics	q_2.kursk russian sinking US submarines

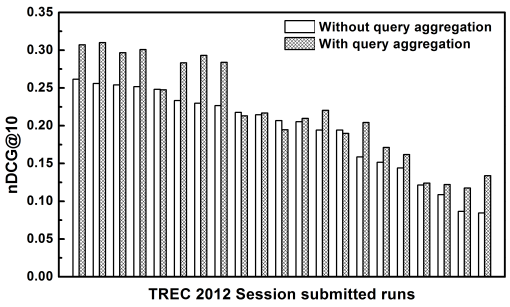

Fig. 1. Search accuracy improvement of the runs which applied query aggregation in TREC 2012 Session track.

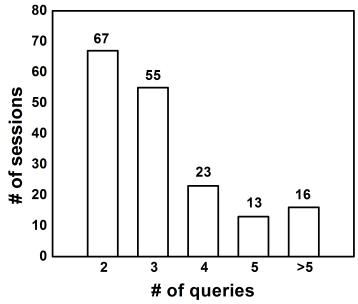

Fig. 2. Session length distribution (174 TREC 2011 and 2012 sessions).

We argue that the importance of queries in a session varies; it is not uniform. Some queries are in fact more interesting to the user; hence it should be treated differently from all other queries. Moreover, a query's importance could be related to the position of the query in the session. For example, an intuitive assumption could be that a query nearing to the last query is more important than others since it motivates the last query.

In this paper, we answer the research question of *"what is the best query aggregation scheme for session search"*. We introduce a new query aggregation scheme and compare it with various existing ones. We propose a three-step aggregation scheme, which is inspired by U-shape query weight curves learned from training data. Through experimenting the new query aggregation scheme in combination with two state-of-the-art retrieval models, we find that the proposed three-step aggregation scheme achieves the best search accuracy for TREC 2012 with a 4.69% absolute gain in nDCG@10 over the TREC 2012 best system.

The remainder of this paper is organized as follows. Section 2 discusses the related work. Section 3 states the problem definition and Section 4 presents the experimental setup. Section 5 describes how we learn the query weight curve by SVM ranking and Section 6 proposes the new three-step query aggregation scheme. Section 7 evaluates various aggregation schemes in combination with different retrieval models. Section 8 concludes the paper.

2 Related Work

Session search is challenging since it involves more queries in search. Existing query aggregation schemes can be grouped into *explicit aggregation* that assigns explicit weights to each query and separates aggregation from the retrieval models and *implicit aggregation* that merges query weighs in the retrieval models.

2.1 Explicit Query Aggregation

We summarize the explicit query aggregation schemes as the following. Let λ_i represents the weight for each query q_i in a session. Below are commonly used query aggregation schemes:

Uniform: All queries are weighted equally with the same weight, i.e.,

$$\lambda_i = 1 \text{ for } i = 1, 2, \cdots, n \tag{1}$$

Previous vs. current (PvC): All queries before the last query share a same lower weight while the last query employs a higher weight:

$$\lambda_i = \begin{cases} \lambda_p & i = 1, 2, \cdots, n-1 \\ 1 & i = n \end{cases} \tag{2}$$

where λ_p denotes the weight of previous queries which satisfies $\lambda_p < 1$. Two top TREC 2012 systems [6] and [11] both used this aggregation scheme. [11] set a discounted weight for the previous queries when computing the Maximum

Likelihood Estimation (MLE) of features such as single terms, ordered, and unordered phrases. [6] directly applied the discounted weights when combining queries. Jiang *et al.* [10, 11] assumed that the last query was most important in a session, although they did not differentiate among previous queries. They assumed that all previous queries were of the same importance, hence they assigned a lower weight to all previous queries and a higher weight to the last query.

First vs. rest (FvR): [14] proposed to reward more to retrieval systems that retrieve more relevant documents in the earlier queries. This suggests higher importance of the first query. Thus, the first query is assigned a higher weight and the other queries including the last query share the same lower weights:

$$\lambda_i = \begin{cases} \lambda_f & i = 1 \\ 1 & i = 2, \cdots, n \end{cases} \tag{3}$$

where λ_f denotes the weight of the first query, which satisfies $\lambda_f > 1$.

Distance-based discounting (Dist.): In TREC Session 2012, [6] used a distance-based weight discounting scheme. They observed that the influence of previous queries and previous search results becomes weaker and weaker. The last query, which is the most novel, is concerned the most by the user, hence the last query receives the most weight. Other queries' weights are reversely correlated to the distance between it to the last query. A monotonically decreasing function can be used to model the weight discounting:

$$\lambda_i = \begin{cases} \frac{\lambda_p}{n-i} & i = 1, 2, \cdots, n-1 \\ 1 & i = n \end{cases} \tag{4}$$

where λ_p is a discount factor for the second last query.

Exponential: [7] implies an exponential discounting function query aggregation. It considers session search as a Markov Decision Process (MDP) [21]. *Queries* are modeled as states, search engines and users are modeled as *agents* in this MDP. The *actions* are divided into *user actions* and *search engine actions*. User actions include keeping, adding, and/or removing query terms from the previous query q_{i-1}. Search engine actions include increasing, decreasing, and maintaining term weights in the scoring functions. The entire session search process can be viewed as the following. Previous queries that the user submitted influence the search results; the search results again influence the user's decision of the next query. This interaction continues until the search stops. Considering the whole session, the overall relevance is scored for document d to a session that starts at q_1 and ends at q_n as:

$$Score(session, d) = \sum_{i=1}^{n} \gamma^{n-i} Score(q_i, d) \tag{5}$$

where $q_1, \cdots, q_i, \cdots, q_n$ are the queries in the session. $\gamma \in (0, 1)$ is the discount factor. The further apart from the last query, the more discounted the query weight by the discount factor γ.

2.2 Implicit Query Aggregation

Implicit query aggregation schemes are also popular. For example, White *et al.* [22] observed that two adjacent queries in a session could be merged by a linear combination. Two queries in a pair were thus assigned complementary weights, less for the former. They assigned a weight with the form of $e^{-(p-1)}$ to a previous query, where p represented how far this query to the last query. Bennett *et al.* [2] chose a different base (0.95) in the above formula.

Liu *et al.* [19] also used query aggregation implicitly in their user behavior based retrieval model. They included all queries to form a term set. When a new query with m terms arrived, each term in the term set was assigned a initial uniform weight of $\frac{1}{m}$. Then the terms of the new query were merged into the term set, by decreasing weights of the terms in the set. This procedure is equivalent to assigning earlier queries less weights.

Because of the complexity of session search, researchers have attempted to first classify sessions and then apply specific aggregation schemes to each class. For example, the CWI group [9] assumed two classes of users: "good" users and "bad" users. A "good" user learns from previous search results in a session to generate a new and high quality query, so that the last query in a session is able to express the information need precisely. On the contrary, a "bad" user fails to learn from the search experience and fails to adjust queries to fit the information need, and end up with more queries in a session. The authors classified the users into "good users" and "bad users" based on the session length. A session submitted by a "good" user receives a discounted weighting scheme that accounts for lower weights for early queries than later queries in a session. The sessions issued by "bad" users received uniform query weights.

It may be a good idea to apply different aggregation schemes according to the session lengths. However, there is no strong evidence to justify that shorter sessions are from "better" users.

Xiang *et al.* [24] also classified sessions. They classed the sessions into four types: reformulation, specialization, generalization, and general association. For sessions of specialization and generalization, they discarded previous queries and only used the last query. For sessions of general association, uniform weights were assigned to each query. They did not apply query aggregation for other types of sessions. This classification of sessions is sensible. However, it is challenging to classify the queries into these types. Therefore, the authors seek for the assistance of external knowledge such as Open Directory Project (ODP)[2]. This might hurt the system efficiency and the practical use of this method in real-time online search systems.

In this paper, we investigate various *explicit* aggregation schemes that are separated from the retrieval models. We report our findings in the following sections.

3 Problem Definition

In this paper, we present the whole session relevance score $Score(session, d)$ as a weighted linear combination of the relevance scores of queries in a session S. Let $Score(session, d)$ denotes the overall relevance score for a document d to a session that begins at q_1 and ends at q_n. The score can be written as:

$$Score(session, d) = \sum_{i=1}^{n} \lambda_i \cdot Score(q_i, d) \tag{6}$$

where λ_i represents the weight for each query $q_i \in S$.

By formulating λ_i differently, we can have different query aggregation schemes. It is worth noting that Eq. (6) allows us to choose aggregation schemes and retrieval models independently.

$Score(q_i, d)$ is the relevance score between document d and the i^{th} query q_i in the session. $Score(q_i, d)$ can be calculated by a wide range of document retrieval models. In this work, In this work, the retrieval models that we use in the experiments include the state-of-the-art language modeling approach as well as a novel model which considers historical search query and results.

In language model, the relevance score $Score(q_i, d) = P(q_i|d)$ is calculated as in [20]:

$$Score(q_i, d) = \sum_{t \in q_i} \log P(t|d) \tag{7}$$

where the probability that t generates d (term weight) $P(t|d)$ is calculated based on multinomial query generation language model with Dirichlet smoothing [25]:

$$P(t|d) = \frac{\#(t, d) + \mu P(t|C)}{|d| + \mu} \tag{8}$$

where $\#(t, d)$ denotes the number of occurrences of term t in document d, $P(t|C)$ is the Maximum Likelihood Estimation (MLE) of the probability that t appears in corpus C, $|d|$ is the document length, and μ is the Dirichlet smoothing parameter (set to 5000).

In the second model, we modify (7) by adapting the weight adjustment for terms in [7]:

$$Score(q_i, d) = \sum_{t \in q_i} w_t \log P(t|d) \tag{9}$$

where

$$w_t = \begin{cases} 1 + \alpha[1 - P(t|d^*_{i-1})] & t \in q_{i-1} \text{ and } t \in q_i \\ -\beta P(t|d^*_{i-1}) + \epsilon idf(t) & t \in q_i \text{ but } t \notin q_{i-1} \\ -\delta P(t|d^*_{i-1}) & t \in q_{i-1} \text{ but } t \notin q_i \end{cases} \tag{10}$$

where $P(t|d^*_{i-1})$ denotes the probability of occurence of term t in SAT clicks [8], $idf(t)$ is the inverse document frequence of term t, α, β, ϵ, and δ are parameters.

Both the language modeling approach and the second model are used in combination with various explicit query aggregation schemes and are tested in this work.

[2] http://www.dmoz.org

Table 2. Dataset statistics for TREC 2011 and TREC 2012 Session tracks.

Year	#topic	#session	#query	Session length
2011	62	76	280	3.68
2012	48	98	297	3.03

Table 3. Training datasets with different session lengths.

Session length	#session	#training instance
2	59	960,485
3	48	840,777
4	20	356,979
5	11	184,359

4 Experimental Setup

In this paper, we use the TREC 2011 and TREC 2012 Session Track [15, 16] datasets in our evaluation. TREC 2011 contains 76 sessions that belongs to 62 topics, while TREC 2012 contains 98 sessions for 48 topics. One TREC topic corresponds to one information need. The users (NIST assessors) search for a given information need such as *You are planning a winter vacation to the Pocono Mountains region in Pennsylvania in the US. Where will you stay? What will you do while there? How will you get there?* (session 7, Table 1). They interacted with a search engine Yahoo! BOSS[3] and created queries in the sessions. Each query triggered and represented an interaction between the search engine and the user. The top 10 returned documents were shown to the users and the user clicked the interesting ones to read. Table 2 lists the statistics of the TREC datasets.

TREC Session tracks emphasize on finding relevant search results for the last query in a session. This setting might not be ideal since it is not guaranteed that the last query is the last query and is a good representation of the whole session information need. In fact, we observe that for some sessions, such as TREC 2012 session 27 – *France won its first (and only) World Cup in 1998. Find information about the reaction of French people and institutions (such as stock markets), and studies about these reactions* – the last query is a bad query. After issuing queries about *stock market* and *1988 world cup* in the first two queries, the user did not find any relevant documents and did not click any retrieved documents. In the last (the third) query, the user issued *french world cup 1998*, which is not a good query since it does not well represent the information need which is about *reaction*. The user might be frustrated and was not willing to nor able to issue a good query in the session.

Nonetheless, we believe that most TREC sessions well represent common Web search behaviors. We have observed interesting patterns through experimenting with TREC datasets. Moreover, the ground truth that TREC Session tracks provide for evaluating search engine accuracy is valuable; which allows us to not only study user search behavior, but more importantly, to study how to make use of the knowledge learned from the search logs for better whole session retrieval. We therefore employ TREC Session data and leave experiments over more popular Web search log datasets, such as the AOL query log and the Yahoo! Webscope query log, as future work.

The corpus for retrieval used in our experiments is the ClueWeb09 CatB[4] collection. Spam documents whose Waterloo's "GroupX" spam score[5] is below 70 [4] are filtered out from the collection. TREC's official ground truth and evaluation scripts are used in the experiments. nDCG@10 is the main evaluation metric used in TREC Session Tracks and also serves as the evaluation metric in our experiments.

As described in Section 3, query aggregation schemes and retrieval models can be applied independently. Therefore, we experiment the aggregation schemes in combination with two retrieval models. The first retrieval model is the language modeling approach. We modify the language modeling implementation in the Lemur search engine[6] to include query aggregation. The second retrieval model is the anonymous model.

5 The U-shape Query Weight Curve

This section reports our findings of the U-shape query weight curve in session search. Initially, we aim to employ a supervised learning to rank framework to obtain the query weights for query aggregation. In this section, we plot the query weight curves that are learned from the training data and our observation and discoveries from them. We then use those learned weights to aggregate the queries in a session for retrieval. However, surprisingly, the learned weights do not generate the best search accuracy as compared to other query aggregation schemes. This might be caused by overfitting. Nonetheless, we believe that we have learned the major trend of query importance in a session. In the next section, we describe the new query aggregation scheme that we derive from these discoveries.

[3] http://developer.yahoo.com/boss/search/
[4] http://lemurproject.org/clueweb09/
[5] http://durum0.uwaterloo.ca/clueweb09spam/
[6] http://www.lemurproject.org/, version 5.0

We employ SVMRank[7] to learn the query weights λ_i. SVMRank aims to optimize a ranking function. If a document d_j is more relevant than another document d_k, the relevance score of d_j is higher than d_k. The ranking function that SVMRank optimizes is in a form of linear combination of features [12]:

$$Score(q,d) = \sum_f w_f \Phi(f,d) \tag{11}$$

where q and d denote query and document respectively, $\Phi(f,d)$ is the score of the document regarding to a feature f, and w_f is the weight of f.

Comparing Eq. (6) and Eq. (11), we substitute w_f and $\Phi(f,d_j)$ in Eq. (11) with λ_i and $Score(q_i,d)$ in Eq. (6). $Score(q,d)$ in Eq. (11) becomes $Score(session,d)$ in Eq. (6); where we calculate the relevance score of a document with regard to the entire session S. For $Score(q_i,d)$ in Eq. (6), we can calculate the score of each document d in the ground truth to q_i using Eq. (7)or Eq. (9). We observe similar trends for the two retrieval models and thus only report one of them.

Table 4. nDCG@10 for query aggregation schemes for Lemur's language model. "↓" means decrement

Aggregation Scheme	TREC 2011		TREC 2012	
	nDCG@10	↓	nDCG@10	↓
Exponential	0.4677	–	0.3288	–
Learning	0.4630	1.00%	0.3259	0.88%

Table 5. nDCG@10 for query aggregation schemes for the second model. "↓" means decrement

Aggregation Scheme	TREC 2011		TREC 2012	
	nDCG@10	↓	nDCG@10	↓
Exponential	0.4821	–	0.3368	–
Learning	0.4816	0.10%	0.3360	0.24%

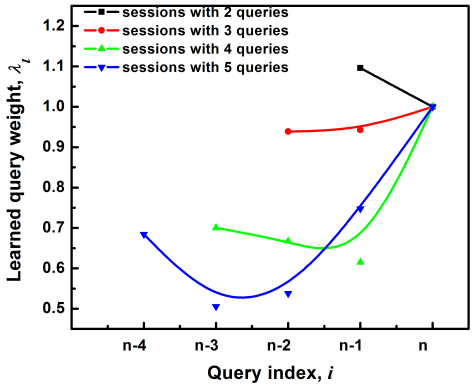

Fig. 3. Learning the weights λ_i for sessions of different length.

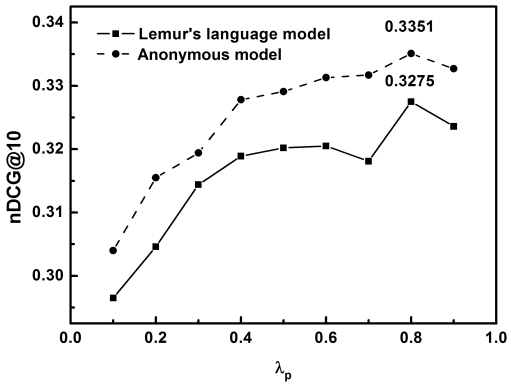

Fig. 4. Tuning of λ_p in the PvC scheme. The value of nDCG@10 reaches the peak at $\lambda_p = 0.8$.

We build a training dataset by merging the ground truth data from both TREC 2011 and TREC 2012 Session tracks. Each document in the TREC ground truth is annotated with one manually assigned score: spam documents with a score of -2, non-relevant 0, relevant 1, highly relevant 2, key relevant 3, and navigational 4. The higher the score, the more relevant a document to the session. This score is used as the manual judgment for the whole session relevance. There are 174 sessions in total and 10-fold cross-validation is used to learn the weights.

Suppose we have 3 documents d_1, d_2, d_3 for session S and they are annotated as $rel(d_1,s) = 1$, $rel(d_2,s) = 4$ and $rel(d_3,s) = -2$, we can calculate $Score(q_i,d_j), j = 1,2,3$, where q_i is the i^{th} query in the session. An example of training instances for SVMRank is shown below:

rel	session	$Score(q_1,d)$	$Score(q_2,d)$	document id
4	1	$1:0.0038$	$2:0.0047$	clueweb09-en0000-15-25059
1	1	$1:0.0055$	$2:0.0069$	clueweb09-en0000-01-01067
-2	1	$1:0.00099$	$2:9.9\times10^{-5}$	clueweb09-en0005-56-32088

where the first column is the human judged relevance score of a document. The third to the second last columns are scores generated by the retrieval models for each query in the session. This example only has two queries in the session. The last column displays the corresponding document identification number.

[7] http://svmlight.joachims.org/

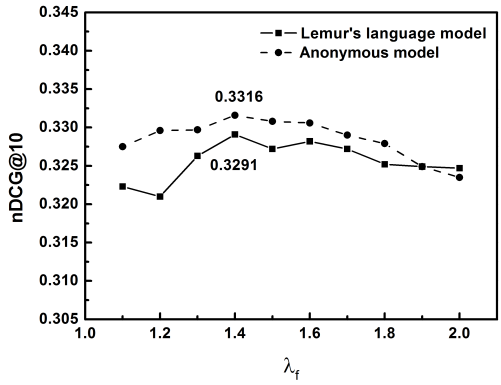

Fig. 5. Tuning of λ_f in the FvR scheme. The value of nDCG@10 reaches the peak at $\lambda_f = 1.4$.

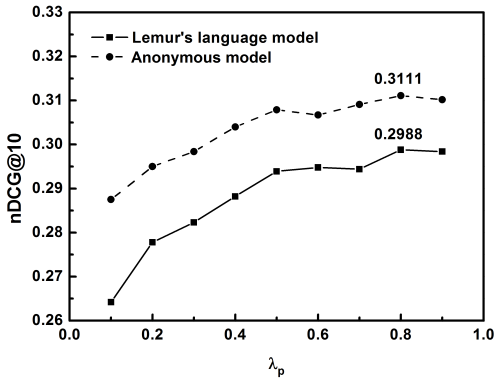

Fig. 6. Tuning of λ_p in the distance-based scheme. The value of nDCG@10 reaches the peak at $\lambda_p = 0.8$.

The sessions are different in terms of their lengths. Figure 2 shows the session length distribution for 174 sessions from TREC 2011 and TREC 2012. We can see that most TREC search sessions contain two or three queries and on average the number of queries per session is 3.31. We group the training sessions with the same number of queries together. For each group, we learn query weights λ_i by a 10-fold cross-validation. Few sessions are longer than five queries, hence we exclude them from the training data. Table 3 lists the number of training instances for sessions in different lengths.

Figure 3 plots the curves of query weight distribution over query position for sessions with lengths of 2, 3, 4, and 5, respectively. The x-axis displays the query position index and the y-axis shows the corresponding query weight. All learned query weights λ_i are normalized by λ_n, the weight for the last query q_n. For example, session 47 has three queries, i.e., $n = 3$, the query weights for q_1, q_2, and q_3 are 0.93, 0.94, and 1.

From Figure 3, we can see that *the learned curves are close to U-shape parabolas*. Moreover, we find that (1) the learned query weights λ_i in general decreases while a query's distance to the last query increases. This indicates that earlier queries have less importance. (2) We also notice that the first query in all sessions demonstrates a higher weight than other queries except the last query. (3) In addition, these U-shapes demonstrate a lower starting point and a higher ending point.

The learned results suggest that besides the (most important) last query, the first query shows a special importance in a session. This may be because that users usually starts a search with the topic words of the information need. Consequently, in many cases, the first query is completely composed of terms that are highly relevant to the topic of a session. It is worth noting that sometimes the first query is even more importance than the last query. It happens when the session length is only two.

We then apply these learned query weights into Eq. (6) and use the formulation to perform document retrieval for the whole session. We compare the search accuracy of the search engine using the learned aggregated queries with using one of the top performing aggregation schemes, exponential discounting scheme. The nDCG@10 evaluation results are shown in Table 4 and 5 for two retrieval methods LM and the anonymous model, respectively. We expect to observe a performance gain in nDCG@10 from the learned query weights as compared to the exponential scheme since the query weights are learned directly from the data. Surprisingly, we find that retrieval models using the learned query weights to aggregate queries experiencing a performance decrease as compared to the exponential discounting scheme.

The undesirable performance of learned query weights is perhaps caused by overfitting. As we can see that the session lengths vary. When we chose a specific weight distribution for each session length, the training data is segmented and becomes more sparse. Moreover, learning and applying different weights for sessions with different lengths might not be infeasible in practice. It is because the last query can be at any where in a session and it is not necessary the last query in a session. We hence have no idea of the entire session length when a user is still issuing queries. Nonetheless, the learned query weight curve reveals interesting patterns and trends in query importance, which inspires us to propose a new query aggregation scheme in the next section.

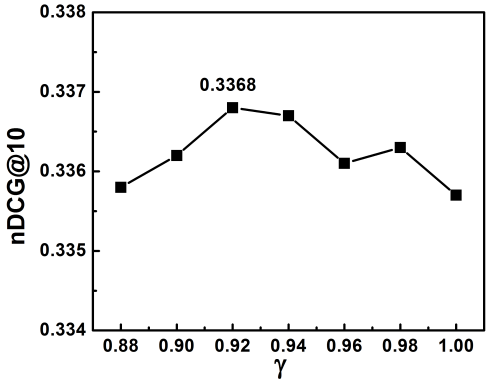

Fig. 7. Tuning of γ in the exponential scheme. The value of nDCG@10 reaches the peak at $\gamma = 0.9$.

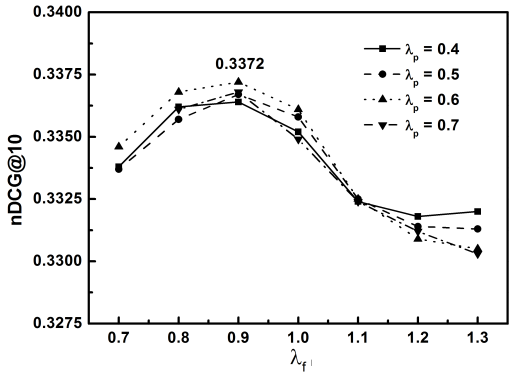

Fig. 8. Tuning of λ_f and λ_p in the three-step scheme. The value of nDCG@10 reaches the peak at $\lambda_f = 0.9$ and $\lambda_p = 0.6$.

6 Three-Step Query Aggregation

In this section, based upon this discoveries that we have made so far, we propose a smoothed version of the learned weights and yield a new three-step query aggregation scheme. The new scheme is one of the best query aggregation schemes in our experiments.

The U-shape query weight curves motivate us to treat the first, the current, and the middle queries in different ways. A three-step function seems a good choice. We thus propose a novel three-step aggregation scheme:

$$\lambda_i = \begin{cases} \lambda_f & i = 1 \\ \lambda_p & i = 2, \cdots, n-1 \\ 1 & i = n \end{cases} \tag{12}$$

where $\lambda_1 = \lambda_f$ denotes the weight of the first query, λ_p denotes the weight of middle queries; they satisfy $0 < \lambda_p < \lambda_f$.

It is worth noting that in most cases $\lambda_1 < \lambda_n$. However, the query weights are not only related to i, the position of the query, but also related to n, the session length. We observe that in Figure 3 that the first query in a short session is of much higher importance than that in a long session. Moreover, the first query looks even more important than the last query in a session that contains two queries. In our opinion, a short session may be able to quickly satisfy the information need, which might be caused by starting with a highly relevant query so that the user no need to modify the query too much. In other words, the first query in a short session is of very high quality and represent the information need well. The higher the weights, the more important the queries, and the better quality of the queries. As a result, the first queries get higher query weights in short sessions than the first queries in long sessions. We explore the optimal values for λ_f and λ_p in Section 7.2.

In addition, from Figure 3, we can also see that shorter sessions have flatter weight curves while the weight curve of longer sessions is relatively steeper. The flat curve suggests a uniform-ish query weighting scheme for short sessions. This conflicts the assumption made by He et al. [9]. They assumed that a user who is satisfied within a short session is good at learning from previous search results and issuing good queries so that the most recently generated queries are most important than the earlier ones. Therefore, they propose to very different weights for individual queries in short sessions. Moreover, they consider that a user who requires a long session to search is not clear on the information need and is a bad learner to learn from previous search results and issue better queries sooner. Therefore, the importance of all queries in the session that are issued by a bad user should not be differentiated; which suggests a uniform distribution of the query weights for the long sessions.

We believe our observations are correct. Short queries usually demonstrate that the user is clearer about the search goal as compared to longer queries. Long queries imply users change their minds and try different queries during search. Therefore, the queries in a shorter session might show less "detour" in search and yield a flatter and higher valued curve for query weights than queries in longer sessions.

7 Evaluation

7.1 Aggregation Schemes under Comparison

We compare the following aggregation schemes in this evaluation. The search algorithm can be either the language modeling system (LM) or the anonymous model.

- Uniform: All queries in a session share the same weight.
- Previous vs. current (PvC): All previous queries $q_1, ..., q_{n-1}$ share a lower weight than the last query as shown in Eq. (2).
- First vs. rest (FvR): The first query has a higher weight and the other queries including the last query share a lower and identical weight as shown in Eq. (3).
- Distance-based (Dist.): Query weights are discounted with how far the query is from the last query. The reciprocal function is used as the discounting function, as shown in Eq. (4).
- Exponential (Exp.): A aggregation scheme proposed by this paper. It is derived from Eq. (5).
- Learning: The weights of queries are learned by SVMRank as described in Section 5.
- Three-step: There are three levels in the query weighting function as defined in Eq. (12). The last query has the highest weight, the intermediate queries share the lowest weight, while the first query has a middle range weight between them.

In addition, the best results of the official TREC 2011 and TREC 2012 Session track (TREC best) [11, 19] are included for comparison and are used as the baseline. They both worked on the ClueWeb09 CatB corpus. Particularly, the system of [19] used an exponential function to discount the weight of query based on the distance between it and the last query. The system of [11] applied an aggregation scheme of the same as PvC, with assigning a weight of 0.6 to last query and a weight of 0.4 to the previous ones.

7.2 Parameter Settings

Query aggregation schemes presented in this paper all contain parameters except the uniform scheme (using 1 for all queries). Since the values of these parameters influence the effectiveness of query aggregation and search accuracy, we conduct parameter selection and present them in this section. All the tuning are done through 10-fold cross-validation.

Table 6. The number of sessions whose first query is completely composed of theme terms.

Year	# of sessions	# of session with theme words as the first query
2011	76	31
2012	98	73

We plot the parameter settings for both the LM and the anonymous retrieval model. For the PvC scheme, we tune λ_p in Eq. (2) over a range of 0.1 to 0.9 to select the best value for λ_p. Figure 4 indicates the trends of nDCG@10 values with λ_p. We find that the curves show the same trends. The nDCG@10 values achieve peaks in both curves at around $\lambda_p = 0.8$. It implies that the aggregation scheme and retrieval model can work independently.

For the FvR scheme, we tune λ_f in Eq. (3) over a range of 1.1 to 2.0. Figure 5 shows plotting of nDCG@10 over the above range. We observe the similar behavior to that of the PvC scheme: the same aggregation scheme show similar trends on different retrieval models, with the peak appearing at the same position around 1.4.

For the distance-based scheme, Figure 6 displays the nDCG@10 values from 0.1 to 0.9. Again, we observe that the nDCG@10 values show similar trends and reach peaks at around $\lambda_p = 0.8$ for both retrieval models.

For the exponential scheme, we plot γ over a range of 0.88 to 0.92. Figure 7 illustrates the relationship between nDCG@10 and γ. nDCG@10 climbs to the peaks near $\gamma = 0.92$. The result suggests that a good discount factor γ is very close to 1. It implies that the previous query contributes nearly the same as the last query and the discount between two adjacent queries should be mild, not too dramatic.

Figure 8 demonstrates the tuning for λ_f and λ_p in the three-step aggregation scheme. We tune λ_p from 0.4 to 0.7 with an interval of 0.1. For each λ_p, we test λ_f from 0.7 to 1.3 with an interval of 0.1, except the values not satisfying $\lambda_f > \lambda_p$. We discover that the curve of $\lambda_p = 0.6$ is higher than other curves, and reaches its peak at $\lambda_f = 0.9$. It is worth noting the relatively high weight of the first query for both retrieval models may imply that the high importance of the first query is independent of the retrieval model.

Table 7. nDCG@10 for various aggregation schemes for Lemur's language model. TREC best serves as the baseline. † indicates a significant improvement over the baseline at $p < 0.05$ (t-test, single-tailed).

Aggregation	TREC 2011		TREC 2012	
Scheme	nDCG@10	%chg	nDCG@10	%chg
TREC best	0.4540	0.00%	0.3221	0.00%
Uniform	0.4475	-1.43%	0.3033	-5.84%
PvC	0.4516	-0.53%	0.3275	1.68%
FvR	0.4390	-3.30%	0.3291	2.17%
Distance-based	0.4271	-5.93%	0.2988	-7.23%
Exp	**0.4677**	**3.02%**†	0.3288	2.08%
Learning	0.4630	1.98%	0.3259	1.18%
3-step	0.4552	0.26%	**0.3340**	**3.69%**†

Table 8. nDCG@10 for various aggregation schemes for the second retrieval model. TREC best serves as the baseline. † indicates a significant improvement over the baseline at $p < 0.05$ (t-test, single-tailed).

Aggregation	TREC 2011		TREC 2012	
Scheme	nDCG@10	%chg	nDCG@10	%chg
TREC best	0.4540	0.00%	0.3221	0.00%
Uniform	0.4626	1.89%	0.3316	2.95%
PvC	0.4713	3.81%†	0.3351	4.04%†
FvR	0.4362	-3.92%	0.3316	2.95%
Distance-based	0.4431	-2.40%	0.3111	-3.42%
Exp	**0.4821**	**6.19%**†	0.3368	4.56%†
Learning	0.4816	6.08%†	0.3360	4.32%†
3-step	0.4781	5.31%†	**0.3372**	**4.69%**†

7.3 Search Accuracy with Language Modeling

Table 7 compares the search accuracy under different aggregation schemes for the language modeling model [25, 20]. TREC participants usually provided high performance systems. Therefore, we use TREC best results as a strong baseline.

We adopt the parameters tuned in Section 7.2 for PvC, FvR, Exponential, and the three-step aggregation schemes. The average learned query weights from Section 5 are used in the Learning aggregation scheme.

From Table 7, we observe that PvC, FvR, Exponential, Learning, and three-step schemes outperform the TREC best runs. Furthermore, the exponential scheme performs the best in TREC 2011 while three-step scheme wins in TREC 2012.

The distance-based scheme performs the worst. In our opinion, the reciprocal function discounts the previous queries too much and hurts the search accuracy.

The proposed three-step scheme improves the search accuracy in TREC 2012. It implies that the first query may be more important than the middle queries because a user usually starts search from the topic words of his information need.

The three-step scheme is not the top in TREC 2011. We investigate the dataset differences between the TREC 2011 and TREC 2012. Table 6 shows the number of sessions whose first query is only composed of theme terms in TREC 2011 and 2012 Session tracks. We find that 74.5% of sessions in TREC 2012 Session track contain the first query composed of only theme terms, while this proportion is much lower in TREC 2011 Session track. This may be the reason why the three-step scheme outperforms the exponential aggregation scheme in TREC 2012 but lose to it in TREC 2011 as shown in Tables 7.

7.4 Search Accuracy with the Second Model

We compare the search accuracy for various aggregation schemes using the anonymous retrieval model. We observe similar trend as in the language modeling approach as well. The results are listed in Table 8.

By comparing Table 7 and Table 8, we find that the retrieval using two different models with query aggregation show similar trends for which aggregation schemes work the best. It again implies that query aggregation may be universally effective and independent of retrieval models.

There are some small differences in the search accuracy of the two retrieval models. First, FvR scheme is slightly better than PvC scheme in LM, while it is worse than PvC scheme in the anonymous model. The reason may lie in the fact that the second retrieval model has already strengthened the first query by calculating the query transition probability for the next query. Therefore, the FvR scheme may over-emphasizes the first query;s importance in the anonymous model. This may also be the reason why three-step scheme improve the search accuracy less significantly in the anonymous model than in LM.

Nonetheless, the threes-step query aggregation scheme statistically significantly outperforms the TREC best systems. It is also the top performing query aggregation scheme for the TREC 2012 Session dataset.

7.5 Plotting the Query Weight Curves

To better understand the behavior of various query aggregation schemes and find out reasons for their different impacts on search accuracy for session search, we plot the query weights for all query aggregation schemes in Figure 9. Our purpose is to lay out all the curves and look into the relationships among the query aggregation schemes.

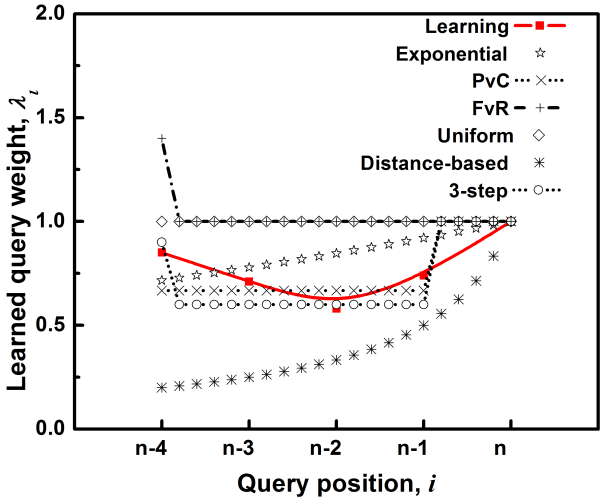

Fig. 9. Query weight curves by various aggregation schemes.

The curve of the averaged learned query weights is the red solid line and is generated as follows. We average λ_i for previous queries over all the subsets with number of queries not greater than 5. For example, $\bar{\lambda}_1$ is averaged by the values of λ_1 trained on the subsets with number of queries of 2, 3, 4, and 5; while $\bar{\lambda}_2$ is averaged by the values of λ_2 trained on the subsets with number of queries of 3, 4, and 5 since λ_2 in the subset with query number of 2 is for the last query. The averaged curve of λ_i against query position is shown in Figure 9 (red line with square symbols). It is thus used as an reference for all other curves. We understand that this learned weight curve did not generate the best search accuracy as we report in earlier section; however, it is directly learned from user data and should be close to real user behaviors. We therefore use it to help understand other aggregation schemes' search accuracy even itself does not directly generate the best performing search accuracy due to overfitting.

We find that the exponential scheme (indicated by the star symbol) aligns the best with the learned weights (the solid square symbols); whereas the distance-based discounting scheme the worst (the cross symbols).

On the other hand, since majority sessions have three or less queries, we focus more on the weights for the last three q_n, q_{n-1}, and q_{n-2}. We find that the exponential scheme aligns almost perfectly with the learned weights at the last two queries and only deviates a bit at q_{n-3}. Since most sessions have only 3 queries, this is a very good match between the learned red query weights for the first 3 queries and the weights generated by the exponential scheme. It might explain why the exponential scheme outperforms other aggregation schemes.

The three-step scheme (indicated by the circle symbol) aligns the learned curve well near the beginning of a session, i.e., the first query. the weight of earliest query, i.e., λ_1, increases again after the continuous decrease of the weights. This might indicate that the first query is also important to the whole session because users often starts with a query full of topic/theme terms, i.e., starting the search from general terms for the information need. The last query in a session receives high weight as well. In addition, it well represents the reduction of query importance for the middle queries. The weights decrease rapidly after q_{n-1}. This might imply that the unit of concept change. In other words, the user does not care about the queries $2 \sim 3$ iterations before when performing a session search. Therefore, in our opinion, this leads to the best performance of the three-step aggregation scheme in TREC 2012 dataset.

8 Conclusion

This paper studies the problem of query aggregation for session search. We first learn the optimal query weights using SVM ranking and discover a U-shape query weight distribution for session with various lengths. After observing the learning results, we propose a novel three-step aggregation scheme. Moreover, by comparing the new query aggregation schemes to the widely used ones such as uniform and distance-based, we conclude that the proposed scheme outperform the existing aggregation schemes. Specially, the three-step aggregation scheme works well for both TREC 2011 and TREC 2012 Session tracks, though it is outperformed by the exponential discounting query aggregation scheme over the TREC 2011 Session data. This suggests a dataset impact on query aggregation schemes.

Through the experiments, we find that the first and the last queries are of more importance to a session. Moreover, for sessions that starts with general theme terms for the sessions, the three-step query aggregation scheme is a better choice since these sessions explicitly require a higher weight to the first query.

The experiments also demonstrate that the query aggregation schemes can impact on the search accuracy independently from the retrieval model. This allows us to flexibly combine various query aggregation schemes and retrieval models in the more general whole session search framework.

References

1. M.-D. Albakour and U. Kruschwitz. University of essex at the trec 2012 session track. In *TREC '12*, 2012.
2. P. N. Bennett, R. W. White, W. Chu, S. T. Dumais, P. Bailey, F. Borisyuk, and X. Cui. Modeling the impact of short- and long-term behavior on search personalization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*.
3. B. Carterette, E. Kanoulas, and E. Yilmaz. Simulating simple user behavior for system effectiveness evaluation. In *Proceedings of the 20th ACM conference on information and knowledge management*.
4. G. V. Cormack, M. D. Smucker, and C. L. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. *Inf. Retr.*, 14(5):441–465, Oct. 2011.
5. B. Croft, D. Metzler, and T. Strohman. *Search Engines: Information Retrieval in Practice*. Addison-Wesley, 1st edition, 2009.
6. D. Guan, H. Yang, and N. Goharian. Effective structured query formulation for session search. In *TREC '12*, 2012.
7. D. Guan, S. Zhang, and H. Yang. Utilizing query change for session search. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*.
8. Q. Guo and E. Agichtein. Ready to buy or just browsing?: detecting web searcher goals from interaction data. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*.
9. J. He, V. Hollink, C. Bosscarino, R. Cornacchia, and A. de Vries. Cwi at trec 2011, session, web, and medical. In *TREC '11*, 2011.
10. J. Jiang, S. Han, J. Wu, and D. He. Pitt at trec 2011 session track. In *TREC '11*, 2011.
11. J. Jiang, D. He, and S. Han. Pitt at trec 2012 session track. In *TREC '12*, 2012.
12. T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02*.
13. R. Jones and K. L. Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM conference on Information and knowledge management*.
14. E. Kanoulas, B. Carterette, P. D. Clough, and M. Sanderson. Evaluating multi-query sessions. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in information retrieval*.
15. E. Kanoulas, B. Carterette, M. Hall, P. Clough, and M. Sanderson. Overview of the trec 2011 session track. In *TREC'11*, 2011.
16. E. Kanoulas, B. Carterette, M. Hall, P. Clough, and M. Sanderson. Overview of the trec 2012 session track. In *TREC'12*, 2012.
17. E. Kanoulas, P. D. Clough, B. Carterette, and M. Sanderson. Session track at trec 2010. In *TREC'10*, 2010.
18. J. Liu and N. J. Belkin. Personalizing information retrieval for multi-session tasks: the roles of task stage and task type. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in Information Retrieval*.
19. T. Liu, C. Zhang, Y. Gao, W. Xiao, and H. Huang. BUPT_WILDCAT at trec 2011 session track. In *TREC '11*, 2011.
20. D. Metzler and W. B. Croft. Combining the language model and inference network approaches to retrieval. *Inf. Process. Manage.*, 40(5):735–750, Sept. 2004.
21. S. P. Singh. Learning to solve markovian decision processes. Technical report, Amherst, MA, USA, 1993.
22. R. W. White, P. N. Bennett, and S. T. Dumais. Predicting short-term interests using activity-based search context. In *Proceedings of the 20th conference on information and knowledge management*.
23. R. W. White, I. Ruthven, J. M. Jose, and C. J. V. Rijsbergen. Evaluating implicit feedback models using searcher simulations. *ACM Trans. Inf. Syst.*, 23(3):325–361, July 2005.
24. B. Xiang, D. Jiang, J. Pei, X. Sun, E. Chen, and H. Li. Context-aware ranking in web search. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*.
25. C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, Apr. 2004.