Information Processing and Management 54 (2018) 252–272

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

# TournaRank: When retrieval becomes document competition

Gilles Hubert*,[a], Yoann Pitarch[a], Karen Pinel-Sauvagnat[a], Ronan Tournier[a], Léa Laporte[b]

[a] IRIT, Université de Toulouse, France
[b] LIRIS, INSA, Lyon, France

ARTICLE INFO

*Keywords:*
IR Model
Feature-based representation
Tournament

ABSTRACT

Numerous feature-based models have been recently proposed by the information retrieval community. The capability of features to express different relevance facets (query- or document-dependent) can explain such a success story. Such models are most of the time supervised, thus requiring a learning phase. To leverage the advantages of feature-based representations of documents, we propose TOURNARANK, an unsupervised approach inspired by real-life game and sport competition principles. Documents compete against each other in tournaments using features as evidences of relevance. Tournaments are modeled as a sequence of matches, which involve pairs of documents playing in turn their features. Once a tournament is ended, documents are ranked according to their number of won matches during the tournament. This principle is generic since it can be applied to any collection type. It also provides great flexibility since different alternatives can be considered by changing the tournament type, the match rules, the feature set, or the strategies adopted by documents during matches. TOURNARANK was experimented on several collections to evaluate our model in different contexts and to compare it with related approaches such as Learning To Rank and fusion ones: the TREC Robust2004 collection for homogeneous documents, the TREC Web2014 (ClueWeb12) collection for heterogeneous web documents, and the LETOR3.0 collection for comparison with supervised feature-based models.

## 1. Introduction

Information retrieval (IR) and ranking are inherently linked. Document ranking implies evaluating the relevance of documents according to a user need, expressed as a query. Pioneering models are based on manually-tunable weighting schemes and matching functions such as BM25, Tf.Idf, or Cosine measure (Manning, Raghavan, & Schütze, 2008). More recent works intend to automatically determine these two facets of IR models. They mainly rely on a feature-based representation of documents. Due to their variety, features used as relevance predictors enable a rich representation of potentially heterogeneous documents. The feature-based works can fit into two main categories: supervised and unsupervised techniques.

On the first hand, supervised approaches, known as Learning To Rank (LTR) methods (Tax, Bockting, & Hiemstra, 2015), have been thoroughly investigated in the literature and often offer very competitive performances provided that enough labelled training data are available (Macdonald, Santos, & Ounis, 2013). On the other hand and to the best of our knowledge, very few works have investigated the use of features in an unsupervised scenario for adhoc retrieval. They have been proposed to fit very particular scenarios using most of the time linear combination of features.

* Corresponding author.
*E-mail address:* hubert@irit.fr (G. Hubert).

https://doi.org/10.1016/j.ipm.2017.11.006
Received 18 October 2016; Received in revised form 22 August 2017; Accepted 14 November 2017
Available online 01 December 2017

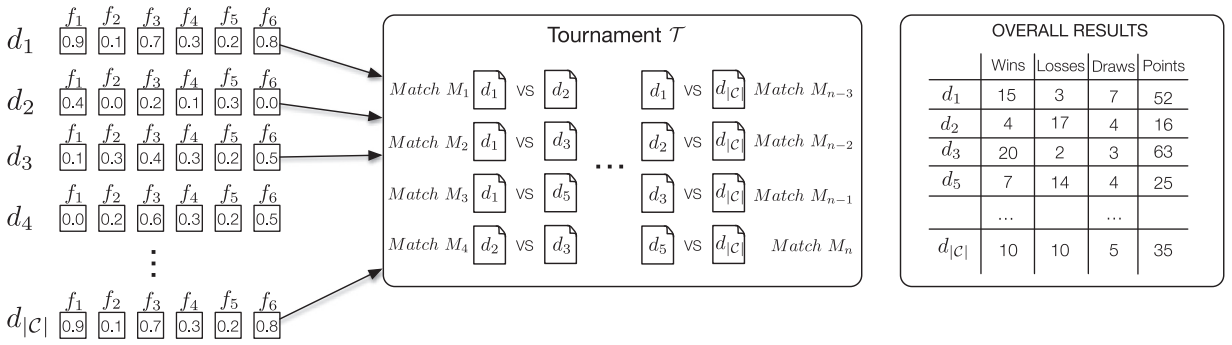



**Fig. 1.** Model overview.

The motivation behind the work presented in this article is twofold: (1) taking advantage of a feature-based representation of documents to leverage on multiple facets of relevance in a generic unsupervised manner, and (2) go beyond the simple query-document matching, by introducing a pairwise document comparison for ranking. The research objective of this article is thus to *explore a new line of research taking benefits from feature-based document representations.* To fulfil this research objective, we propose TOURNARANK, an unsupervised model drawing inspiration from principles of real-life games and sport competitions. Documents, represented by a set of features, oppose each other and gain points during matches organized in tournaments. The features are used as attack and defense characteristics during matches. Features can be for instance the PageRank, query term frequencies or retrieval scores according to common IR models. Documents are then ranked according to the final score board of the tournament. To the best of our knowledge, this is the first method in the literature based on these principles. The main strength of TOURNARANK is its high flexibility, since one can tweak (1) the document representation, i.e., the set of features considered for describing the document; (2) the tournament type (possibly considering an initial ranking); and (3) how documents compete against each other. We designed extensive experiments intended to give answers to various questions related to the concepts underlying our model, i.e., features, matches, and tournaments. Several collections are used in order to evaluate our model in different contexts (homogeneous vs heterogeneous documents and supervised vs unsupervised approaches): the TREC Robust2004, the TREC Web2014 (ClueWeb12), and the LETOR3.0 collections.

The rest of this article is organized as follows. Our ranking model is defined in Section 2. TOURNARANK applied as a reranking model is presented in Section 3. The experimental setup is described in Section 4. The experiment results are detailed and discussed in Section 5. Section 6 presents the related work and Section 7 finally concludes the article.

## 2. TOURNARANK model

Before formally defining TOURNARANK, we first illustrate the tournament principle through an example given in Fig. 1. Given a set of documents $\mathscr{C}$, each document $d_i \in \mathscr{C}$ is represented by 6 features, denoted by $f_1$ to $f_6$. A subset of $\mathscr{C}$ is qualified to participate to the tournament. In our example, a document is qualified if $f_1 > 0$. During the tournament, a sequence of matches between two documents is organized. During a match between $d_i$ and $d_j$, both documents play in turn their features. The better the feature values, the higher the chance to win the match. After the match, both documents earn points depending on their result (win, loss, or draw). In the example, a document earns 3 points if it wins, 1 point in case of a draw, and 0 point if it loses the match. At the end of the tournament, the total number of points each document has earned is considered to rank documents. In our example, $d_3$ is ranked before $d_1$, $d_{|\mathscr{C}|}$ is ranked before $d_5$ and $d_2$.

### 2.1. Model concepts and preliminary notation

Let us consider a collection of documents $\mathscr{C}$, a set of features $\mathscr{F} = \{f_i\}$, and a query $q$. $\mathscr{C}_\mathscr{T}$ is the set of qualified documents (according to $q$) for a tournament $\mathscr{T}$. In this article, documents are qualified according to the values of a given feature. A tournament $\mathscr{T}$ is a sequence of matches between pairs of documents, i.e., $\mathscr{T} = \langle M(d_1, d_2), M(d_3, d_7), ...\rangle$. As in sport competitions, many types of tournaments can be designed, e.g., soccer league or chess tournaments. Those used in the TOURNARANK framework are described in Section 2.2. A match $M(d_i, d_j)$ brings together two documents, $d_i$ and $d_j$, according to a set of rules described in Section 2.3.2. Each document $d$ is represented as a vector of feature values , denoted by $f^d$, defined over $\mathscr{F}$. Before each match, each document $d$ is assigned the same initial life gauge, denoted by $\Lambda$, which decreases during the match according to the different strikes. When playing a match $M(d_i, d_j)$, the document $d_i$ (resp. $d_j$) follows a strategy, denoted by $\sigma_{d_i}$ (resp. $\sigma_{d_j}$) which consists in ordering its features into a list. This list as well as the adopted strategy will be indistinguishably denoted by $\sigma_d$ in the

**Table 1**
Notation used throughout the article.

| Notation | Definition |
|---|---|
| $\mathscr{C}$ | The collection of documents |
| $q$ | The query for which a ranking is produced |
| $\mathscr{T}$ | The tournament associated with query $q$ |
| $\mathscr{C}_{\mathscr{T}}$ | The set of documents qualified for $\mathscr{T}$ |
| $\mathscr{F} = \{f_i\}$ | The set of features |
| $f^d$ | The feature vector of a document $d$ over $\mathscr{F}$ |
| $f_i^d$ | The value of the feature $f_i$ of $d$ |
| $\varphi_i(x, y)$ | The distance function defined over the $f_i$ values |
| $\preceq_i$ | The order relation defined over the $f_i$ values |
| $M(d_i, d_j)$ | A match between documents $d_i$ and $d_j$ |
| $pw$ | The number of credited points in case of a win |
| $pd$ | The number of credited points in case of a draw |
| $(p_i, p_j)$ | The result of a match (the points credited to each document) |
| $\sigma^{(\cdot)}$ | A document strategy type |
| $\sigma_d$ | The strategy of document $d$ following the strategy type $\sigma^{(\cdot)}$ |
| $w_d$ | The number of points a document $d$ is credited during a tournament |
| $\Lambda$ | The initial life gauge of documents |
| $\lambda_d$ | The life gauge of a document $d$ during a match |
| $\mathscr{T}^{RR}$ | Round Robin tournament with no boost factor |
| $\mathscr{T}^{SS}(r)$ | Swiss System tournament with $r$ rounds, and no boost factor |
| $\mathscr{T}^{PRR}(\rho, N)$ | Round Robin tournament with document pooling (ρ pools), selecting the top $N$% of each pool to compete in the final stage, and no boost factor |
| $\mathscr{T}^{PSS}(r, \rho, N)$ | Swiss System tournament with $r$ rounds, document pooling (ρ pools), selecting the top $N$% of each pool to compete in the final stage, and no boost factor |
| $\mathscr{T}^{(\cdot)u}(\alpha)$ | A tournament $\mathscr{T}^{(\cdot)}$ with upper weighting and boost factor $\alpha$ |
| $\mathscr{T}^{(\cdot)s}(\alpha, X)$ | A tournament $\mathscr{T}^{(\cdot)}$ with seed weighting, boost factor $\alpha$, and document being boosted if it wins over a top X% document in the initial ranking |

rest of this article. The strategies a document can adopt are described in Section 2.3.3. The result of a match is a pair of integers $(p_i, p_j)$ such that $p_i$ (resp. $p_j$) represents the number of points awarded by document $d_i$ (resp. $d_j$). The number of points a document is awarded in case of a win (resp. a draw) is denoted by $pw$ (resp. $pd$). A match can thus formally be represented by a function $M$: $\mathscr{C}_{\mathscr{T}} \times \mathscr{C}_{\mathscr{T}} \rightarrow \{pw, pd, 0\} \times \{pw, pd, 0\}$.

During a tournament $\mathscr{T}$, documents accumulate points depending on the results of the matches they are involved in. The total amount of points a document $d$ is awarded during a tournament $\mathscr{T}$, is denoted by $w_d$. Once the tournament is over, i.e., all the matches were played, documents are ranked according to their $w_d$ scores. Notation presented in this section as well as other notation presented below are summarized in Table 1.

### 2.2. Tournament definition

We now provide a description of different tournament types usually found in sport competitions and used in TOURNARANK.

#### 2.2.1. Round Robin tournament

Given a query $q$, the *Round Robin tournament*, denoted by $\mathscr{T}^{RR}$ is inspired by very well-known sport competitions such as soccer championships. In this type of tournament, each participant, i.e., a document $d_i$, plays against all the other participants $d_j \in \mathscr{C}_{\mathscr{T}}$. The amount of points $w_{d_i}$ a document $d_i$ is awarded at the end of a tournament $\mathscr{T}^{RR}$ is calculated as follows:

$$w_{d_i} = \sum_{d_j \in \mathscr{C}_{\mathscr{T}} - \{d_i\}} p_i, \quad \text{with } (p_i, p_j) = M(d_i, d_j)$$

where $p_i$ is the number of points awarded to the document $d_i$ during its match against $d_j$.

#### 2.2.2. Swiss system tournament

The *Swiss System tournament* is inspired by chess tournaments[1] and aims at determining winners reliably, by performing less matches than the Round Robin tournament. Indeed each competitor does not play all the other competitors but only those belonging to its group (groups gather documents having the same number of points). Matches are scheduled one round at a time between competitors who have a similar current cumulative score, after each round of the tournament. Algorithm 1 describes the Swiss System

[1] https://en.wikipedia.org/wiki/Swiss-system_tournament.

**Input**: $C_{\mathcal{T}}, r$
**Output**: $C_{\mathcal{T}}$ ordered by decreasing document score
1 n ← 0 ;
2 **while** $n < r$ **do**
3 $\quad \mathcal{G} \leftarrow$ Partition $C_{\mathcal{T}}$ into groups s.t. $\quad w_{d_i} = w_{d_j} = w, \forall \{d_i, d_j\} \subseteq G_w$ with $G_w \in \mathcal{G}$;
4 $\quad$ **foreach** $G_w \in \mathcal{G}$ *in descending order w.r.t.* $w$ **do**
5 $\quad\quad P \leftarrow$ Pairing($G_w$);
6 $\quad\quad$ **foreach** $(d_i, d_j) \in P$ **do**
7 $\quad\quad\quad (p_i, p_j) \leftarrow M(d_i, d_j)$;
8 $\quad\quad\quad w_{d_i} \leftarrow w_{d_i} + p_i$;
9 $\quad\quad\quad w_{d_j} \leftarrow w_{d_j} + p_j$;
10 $\quad$ n← n+1;

**Algorithm 1.** Swiss-System tournament algorithm.

tournament principles in our IR context. The number of rounds $r$ is predefined. The first step of each round is grouping documents (line 3) into the group partition $\mathcal{G}$. Grouping is performed w.r.t. the current sum of awarded points per document, $w$. Once groups are established, the pairing function (line 5) is called in descending order according to the number of points. Pairing is applied on groups composed of documents having the same amount of points, and proceeds as follows. Given a group $G_w \subseteq \mathcal{G}$, a graph with all documents in $G_w$ is created. Documents that have not played yet are connected. A Blossom-based algorithm (Edmonds, 1965) is used to compute a maximal matching of the graph. Any unpaired document is added to the group with the next highest total number of points. The process is repeated until there are either one or zero unpaired documents. If there is one unpaired document, a *bye* is assigned and it will play during the next round. Each document can thus play at most one match per round. Once pairing $P$ is performed, i.e., the matches are scheduled, matches are run (line 7) and awarded points are added (lines 8–9) to the current score $w_{d_i}$ (resp. $w_{d_j}$) of the document $d_i$ (resp. $d_j$). This process is repeated until the number of rounds $r$ is reached. Swiss System tournaments with $r$ rounds are denoted by $\mathscr{T}^{SS}(r)$ in the rest of the article.

**Remark.** A possible drawback of this tournament type is that, although top and bottom competitors are reliably determined with fewer rounds than with a Round Robin, the soft underbelly may be less reliable. However, this drawback is negligible if we focus on metrics evaluating the top-ranked documents.

### 2.2.3. Two-stage tournaments

A two-stage tournament strategy adds a selection step to better identify documents that will be top-ranked. As a side-effect, this interestingly limits the number of matches and thus improves efficiency.

In a first stage, documents are assigned to ρ pools. The allocation is based on the feature that serves for the tournament qualification to produce similar pools. The set of documents is split in three subsets (top-ranked, mid-ranked, and bottom-ranked documents according to the qualifying feature). The documents belonging to each subset are randomly distributed to produce homogeneous pools. A (sub-)tournament is then run within each pool.[2] In a second stage, the top $N$% documents with highest scores per pool are selected and an ultimate (sub-)tournament is launched on these documents. Documents are eventually ranked according to their final scores. Round Robin and Swiss System tournaments using document pooling are respectively denoted by $\mathscr{T}^{PRR}(\rho, N)$ and $\mathscr{T}^{PSS}(r, \rho, N)$.

### 2.2.4. Analysis of tournaments

From an efficiency point of view, the number of played matches can be critical. Given a tournament $\mathscr{T}^{(\cdot)}$ with $|\mathscr{C}_{\mathscr{T}}|$ documents, $r$ rounds, ρ pools and $N$% documents selected per pool, the function $f(\mathscr{T}^{(\cdot)}, |\mathscr{C}_{\mathscr{T}}|, r, \rho, N)$ that evaluates the number of matches is:

$$\begin{cases} C^2_{|\mathscr{C}_{\mathscr{T}}|} & \text{if } \mathscr{T}^{(\cdot)} = \mathscr{T}^{RR} \\ r \cdot \left( \dfrac{|\mathscr{C}_{\mathscr{T}}|}{2} \right) & \text{if } \mathscr{T}^{(\cdot)} = \mathscr{T}^{SS}(r) \\ \rho \cdot C^2_{\frac{|\mathscr{C}_{\mathscr{T}}|}{\rho}} + C^2_{N \cdot |\mathscr{C}_{\mathscr{T}}|} & \text{if } \mathscr{T}^{(\cdot)} = \mathscr{T}^{PRR}(\rho, N) \\ \rho \cdot r \cdot \dfrac{|\mathscr{C}_{\mathscr{T}}|}{2\rho} + r \cdot \dfrac{|\mathscr{C}_{\mathscr{T}}|}{2} \cdot N & \text{if } \mathscr{T}^{(\cdot)} = \mathscr{T}^{PSS}(r, \rho, N) \end{cases}$$

[2] A sub-tournament is a tournament as defined previously, but run only over a subset of documents in $\mathscr{C}_{\mathscr{T}}$.

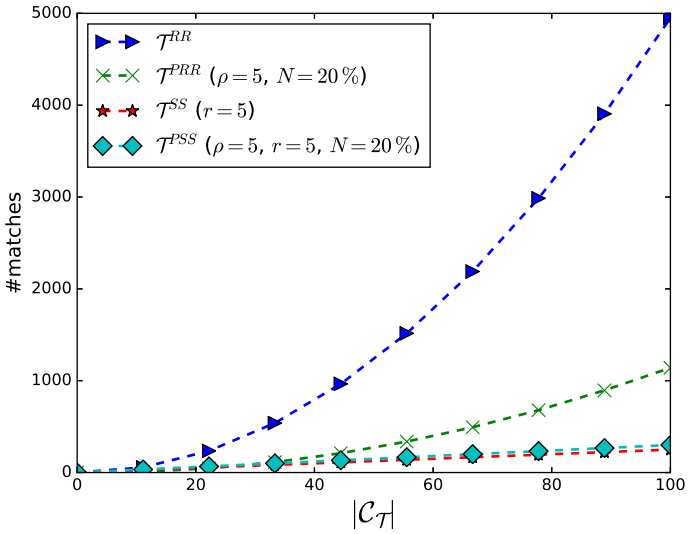



**Fig. 2.** Number of matches w.r.t. the tournament type.

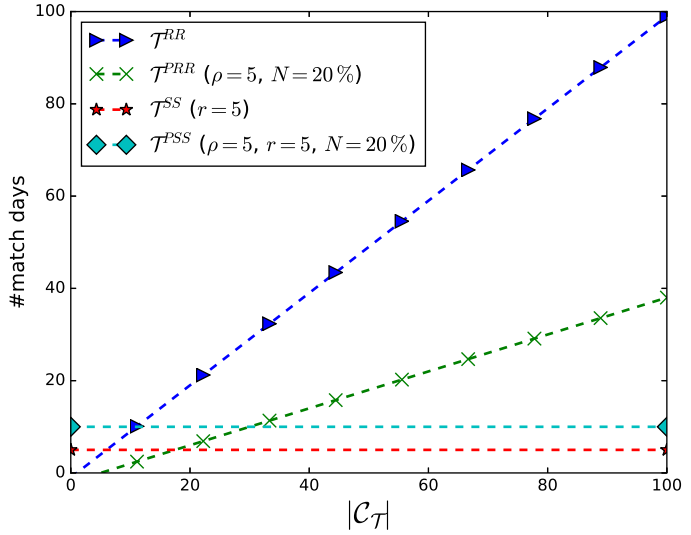



**Fig. 3.** Number of match days w.r.t. the tournament type.

When associated with *two-stage* tournaments, the function $f$ is decomposed into two operands: the first one is dedicated to determine the number of matches during the pooling stage while the second operand determines the number of matches during the final tournament. In practice, for $\mathscr{T}^{PSS}$, the function $f$ can be reduced to $(N + 1)\cdot r\cdot\frac{|\mathscr{C}_{\mathscr{T}}|}{2}$. From these functions, it can be seen that the size of $\mathscr{C}_{\mathscr{T}}$ plays a crucial role in the number of played matches. The behavior of these functions when varying the size of $\mathscr{C}_{\mathscr{T}}$ is reported in Fig. 2. As expected, the number of matches is much lower for Swiss tournaments than for Round Robin ones. It should be noted that the number of matches for Swiss tournaments is the worst case, because if a document cannot be paired during a round it does not play in this round. However, it is the exact number of played matches for Round Robin tournaments. The pooling strategy inherently introduces a last round in the tournament for the best documents. From an efficiency point of view, it is yet very beneficial to $\mathscr{T}^{PRR}(\rho, N)$ since it drastically reduces the number of matches in each tournament that uses pools. However, the number of matches remains unchanged during the first stage of Swiss System

tournaments that use pooling, which explains the lack of interest for pooling documents with Swiss System tournaments. We report execution times in Section 5.

**On the parallelization of tournaments.** Efficiency can be again improved by considering that many matches can actually be run in parallel. We thus introduce the concept of match days as in soccer championships, i.e., a document plays at most one match during a match day. Fig. 3 displays the theoretical number of match days w.r.t. the tournament type.

Comparing Figs. 2 and 3, and assuming that the execution times of a match and a match day are comparable with parallelization, i.e., all the matches in a match day can be run simultaneously, one may infer that efficiency can be improved by at least ten times. For instance, if we consider $\mathscr{T}^{RR}$ tournaments and $|\mathscr{C}_{\mathscr{T}}|=100$, the execution time would be shortened from 5000 to 100.

### 2.3. Match definition

This section starts by discussing how feature values can be compared, then details the match rules, and finally presents the different strategies a document can use to compete against another document.

#### 2.3.1. Feature value comparison

During matches, documents fight each other by comparing their feature values. It is thus necessary that, given a feature $f_i$, their values are comparable. In this section, we introduce two key concepts associated to each feature: an order function, denoted by $\preceq_i$, to determine which document wins on a given feature, and a distance function, denoted by $\varphi_i$, to determine the impact of each strike.

In this work, it is assumed that, for each feature $f_i$, it exists a single optimal value conveying relevance, denoted by $\triangle_i$, and a minimal one, denoted by $\triangledown_i$. Both the order and the distance functions depend on the position of $\triangle_i$ in the distribution of the feature values. Three cases should be considered:

Case 1 $\triangle_i$ is the highest value of the feature value distribution. This is typically the case when considering $f_i$ being the cosine value w.r.t. the query where the highest possible value is 1 and means a perfect fit between the document and the query, i.e., $\triangle_i = 1$. In this scenario, given two documents $d$ and $d'$, we have $d \preceq_i d'$ if $f_i^d \leq f_i^{d'}$ and $\varphi_i(f_i^d, f_i^{d'}) = \frac{\left|f_i^d - f_i^{d'}\right|}{std(f_i)}$. Here, we divide the difference between the two feature values by the standard deviation of $f_i$ to smooth the various feature distributions.

Case 2 $\triangle_i$ is the lowest value of the feature value distribution.This is typically the case when considering $f_i$ being a distance to the query, e.g., the Euclidean distance, where the lowest possible value is 0 and means a perfect fit between the document and the query, i.e., $\triangle_i = 0$. In this scenario, given two documents $d$ and $d'$, we have $d \preceq_i d'$ if $f_i^d \geq f_i^{d'}$ and $\varphi_i(f_i^d, f_i^{d'}) = \frac{\left|f_i^d - f_i^{d'}\right|}{std(f_i)}$.

Case 3 $\triangle_i$ is neither the highest nor the lowest value of the feature value distribution. This is typically the case when considering $f_i$ being the dwell-time (Liu & Belkin, 2015). In this scenario, given two documents $d$ and $d'$, we have $d \preceq_i d'$ if $|f_i^d - \triangle_i| \geq |f_i^{d'} - \triangle_i|$ and $\varphi_i(f_i^d, f_i^{d'}) = ||f_i^d - \triangle_i| - |f_i^{d'} - \triangle_i||$.

It may happen that given a document $d$, its value for feature $f_i$ is not available. In this case, we hypothetize that $f_i^d$ is set to $\triangledown_i$, i.e., $\forall\, d'\ f_i^d \preceq f_i^{d'}$.

#### 2.3.2. Match rules

In a match, two documents face each other and play in turns their features. The general match procedure is presented in Algorithm 2. Initially, both documents start the match with their life gauges, denoted by $\Lambda$, at the same level (lines 3–4). Each document $d$ ranks its features according to the strategy type denoted by $\sigma^{(\cdot)}$, which basically consists in ordering its features (document strategies are further described in Section 2.3.3) (lines 1–2). This results in the strategy followed by the document $d$, denoted by $\sigma^d$.

The first document to strike is randomly chosen (line 5). When its turn comes, the document pops its next top remaining feature from its strategy (line 8 or 12). The *Strike* function is then called (Algorithm 3) where the other document gets its value of the same feature ($f_i$) and removes it from its document strategy (line 2). Both values are compared (line 2) and the life gauge of the document with the lowest value according to the order relation $\preceq_i$ is then reduced by the distance $\varphi_i$ between the two feature values.

This process is repeated until either all features have been played or one document's life reached 0. At the end, the winner is the document having the highest life. Each win yields a gain of *pw* points for the winning document and 0 for the looser. In case of a draw, both documents yield a gain of *pd* points.

Let us illustrate these match rules using the documents in Fig. 4. In this example, each feature takes its values in the [0, 1] range , where 1 is the optimal value (case 1 in Section 2.3.1). Both documents $d_1$ and $d_2$ start the match with a life gauge $\Lambda = 3$ and with features ordered as shown in Fig. 4 (see strategies described in Section 2.3.3). For ease of reading, we assume that, $\forall\, f_i,\ \ std(f_i) = 0.5$.

**Input**: $\Lambda, d_i, d_j, \sigma^{(\cdot)}$
**Output**: $(p_i, p_j)$
$\sigma_{d_i} \leftarrow$ getStrategy$(d_i, \sigma^{(\cdot)})$;
$\sigma_{d_j} \leftarrow$ getStrategy$(d_j, \sigma^{(\cdot)})$;
$\lambda_{d_i} \leftarrow \Lambda$;
$\lambda_{d_j} \leftarrow \Lambda$;
currentPlayer $\leftarrow$ rand$(d_i, d_j)$;
**while** $(\lambda_{d_i} > 0$ *and* $\lambda_{d_j} > 0)$ *and* $(|\sigma_{d_i}| > 0$ *or* $|\sigma_{d_j}| > 0)$ **do**
  **if** *currentPlayer* $= d_i$ **then**
    $f_{played} \leftarrow pop(\sigma_{d_i})$;
    $Strike(\lambda_{d_i}, \lambda_{d_j}, \sigma_{d_j}, f_{played})$;
    **if** $|\sigma_{d_j}| > 0$ **then** *currentPlayer* $\leftarrow d_j$;
  **else**
    $f_{played} \leftarrow pop(\sigma_{d_j})$;
    $Strike(\lambda_{d_j}, \lambda_{d_i}, \sigma_{d_i}, f_{played})$;
    **if** $|\sigma_{d_i}| > 0$ **then** *currentPlayer* $\leftarrow d_i$;
**if** $\lambda_{d_i} > \lambda_{d_j}$ **then**
  **return** $(pw, 0)$;
**else if** $\lambda_{d_j} > \lambda_{d_i}$ **then**
  **return** $(0, pw)$;
**else**
  **return** $(pd, pd)$;

**Algorithm 2.** $M(\Lambda, d_i, d_j, \sigma^{(\cdot)})$.

**Input**: $\lambda_X, \lambda_Y, \sigma_Y, f_i$
**Output**: $\lambda_X, \lambda_Y, \sigma_Y$
$\sigma_Y \leftarrow \sigma_Y - \{f_i\}$;
**if** $f_i^Y \preceq_i f_i^X$ **then**
  $\lambda_Y \leftarrow \lambda_Y - \varphi_i(f_i^X, f_i^Y)$;
**else**
  $\lambda_X \leftarrow \lambda_X - \varphi_i(f_i^X, f_i^Y)$;

**Algorithm 3.** *Strike*$(\lambda_X, \lambda_Y, \sigma_Y, f_i)$.

Document $d_1$ is randomly chosen to start the match and strikes $d_2$ using feature $f_1$ (which is its "best" feature). As $f_1^{d_1}$ (=0.9) is greater than $f_1^{d_2}$ (=0.4), we thus have after the round $\lambda_{d_1} = 3$ and $\lambda_{d_2} = 3 - \left(\frac{0.9-0.4}{0.5}\right) = 2$. $f_1$ is then removed from both document strategies, and $d_2$ continues playing using $f_5$, resulting with the following values of λ: $\lambda_{d_1} = 3 - \left(\frac{0,3-0,2}{0.5}\right) = 2.8$ and $\lambda_{d_2} = 2$. $d_1$ then uses feature $f_6$ (which does not exist for $d_2$) and we have $\lambda_{d_1} = 2.8$ and $\lambda_{d_2} = 2.5 - \left(\frac{0.8-0}{0.5}\right) = 0.9$. $d_2$ then uses feature $f_3$ and we have $\lambda_{d_1} = 2.8$ and $\lambda_{d_2} = 0.9 - \left(\frac{0.7-0.2}{0.5}\right) = -0.1$. Since $\lambda_{d_2} < 0$, i.e., $d_2$ is *dead*, the document $d_1$ wins and gets *pw* points.

### 2.3.3. Document strategies

During a tournament, all the documents adopt the same type of strategy, denoted by $\sigma^{(\cdot)}$. Intuitively, the strategy a document $d$ adopts is the key element in determining how victorious it can be. There exists $|\mathscr{C}_{\mathscr{F}}|!$ potential strategies. A naive one, named the *order-by-value strategy* and denoted by $\sigma^{v}$, consists in playing features in descending order of feature values.

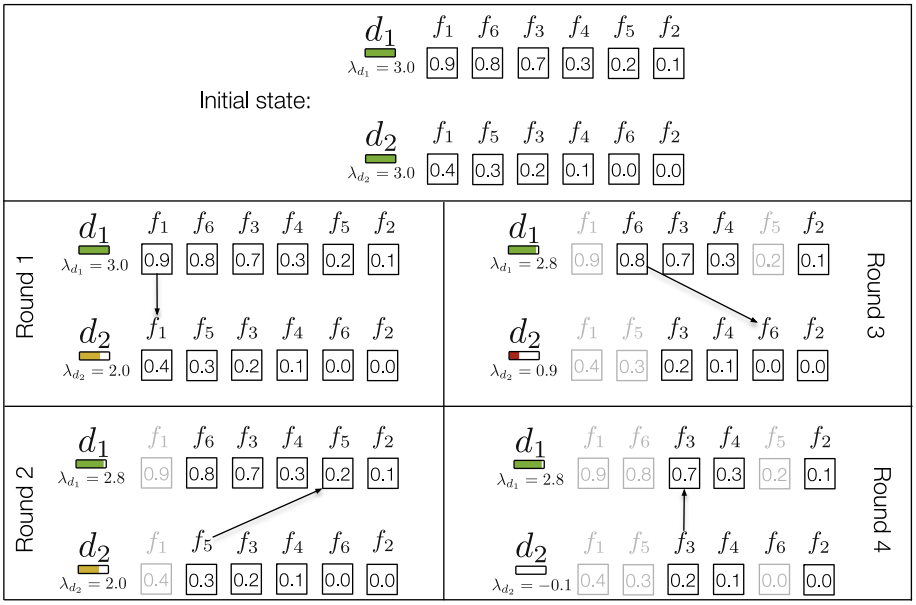



**Fig. 4.** Match example with $\Lambda = 3.0$.

Formally, let $\sigma_d^v$ be the *order-by-value strategy* played by document $d$, while $f_i$ and $f_j$ are two features in $\sigma_d^v$. Feature $f_i$ is played before feature $f_j$ if $f_i^d > f_j^d$. If $f_i^d = f_j^d$, $f_i$ is played first if $i < j$. For instance, in Fig. 4, both $d_1$ and $d_2$ use the *order-by-value strategy*. Another intuitive strategy, named *order-by-rank strategy* and denoted by $\sigma^r$, is to order the features according to the ranking of the document within the features. In case of ties in ranking values, the associated features are randomly chosen. For instance, let us consider the documents $d_1$ and $d_2$ from the Fig. 4, $|\mathscr{G}_\mathscr{F}| = 10$, and Table 2 which gives for each feature $f_i$ the rank of documents $d_1$ and $d_2$ according to their associated order relation $\preceq_i$. The *order-by-rank strategy* for $d_1$ is $f_1, f_4, f_6, f_3, f_2, f_5$ and for $d_2$ is $f_5, f_3, f_1, f_4, f_6, f_2$.

Supervised strategies can also be elaborated. Some of them were experimented, e.g., ranking features according to their entropy, Gini index, or NDCG. Since results were very similar to the unsupervised strategies, they are not presented. We leave for future work strategy adaptation during a match or a tournament.

### 2.4. Discussion

At this point, it is worth positioning our model w.r.t. the state of the art. First, TOURNARANK may be related to the fusion methods if one views each feature as a list generator. In both cases, the order induced by feature values is essential for the final ranking of documents. Second, as Learning To Rank (LTR) approaches, our model relies on a feature-based representation of documents. However, as opposed to LTR approaches TOURNARANK is fully unsupervised. Third, as only a subset of documents is qualified to participate to the tournament and as this qualification may depend on an initial ranking that may be used during matches, TOURNARANK can fall into the reranking category of approaches as detailed in Section 3. Finally, although documents fight each other in different games, TOURNARANK does not rely on game theory (Zhai, 2015).

## 3. TOURNARANK as a reranking model

As aforementioned, TOURNARANK can be easily used as a reranking model. Indeed, given a query, a tournament can be run on an already ranked set of documents. In this case, the list to be reranked is the one induced by the feature used in the qualification step ($f_1$ in the example of Fig. 1). In other terms, $\mathscr{G}_\mathscr{F}$ is considered as an ordered set, such that $d_i > d_j$ if $d_i$ is better ranked than $d_j$ according to the qualifying feature.

One can argue that the initial ranking should be taken into account in the match results. Particularly, an advantage should be given to the documents that win against an initially better-ranked document. This can be done by introducing a boosting factor $\alpha > 1$ that favors initially low-ranked documents against higher-ranked ones. This boosting factor can be used in two different

**Table 2**
Document rankings according to the features $f_1$ to $f_6$ for $d_1$ and $d_2$.

| Feature | | | | | | |
|---|---|---|---|---|---|---|
| Document | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ |
| $d_1$ | 2 | 7 | 4 | 2 | 7 | 3 |
| $d_2$ | 8 | 10 | 7 | 8 | 5 | 9 |

**Input**: $\Lambda, d_i, d_j, b, \alpha$
**Output**: $(p_i, p_j)$

1 *Execute lines 1 to 14 of Algorithm 2. Then:*
2 **if** $\lambda_{d_i} > \lambda_{d_j}$ **then**
3  **if** $b = upper$ *and* $d_j > d_i$ **then**
4  **return** $(\alpha * pw, 0)$;
5  **else if** $b = seed$ *and* $d_i$ *in the top* $X\%$ **then**
6  **return** $(\alpha * pw, 0)$;
7  **else**
8  **return** $(pw, 0)$;
9 **else if** $\lambda_{d_j} > \lambda_{d_i}$ **then**
10  **if** $b = upper$ *and* $d_i > d_j$ **then**
11  **return** $(0, \alpha * pw)$;
12  **else if** $b = seed$ *and* $d_j$ *in the top* $X\%$ **then**
13  **return** $(0, \alpha * pw)$;
14  **else**
15  **return** $(0, pw)$;
16 **else**
17  **return** $(pd, pd)$;

**Algorithm 4.** $M(\Lambda, d_i, d_j, b, \alpha)$.

scenarios: *Upper boost* and *Seed boost*. In the *Upper boost* scenario the number of points awarded to the winner is boosted if it is initially lower ranked than its opponent. In the *Seed boost* scenario the number of points awarded to the winner is boosted if its opponent is ranked in the top $X$% of the initial ranking. To fit this boosting definition, Algorithm 2 has to be modified into Algorithm 4. In the rest of the article, a tournament with no boost is denoted by $\mathscr{T}^{(\cdot)}$ (with $(\cdot) \in \{RR, SS, PRR, PSS\}$), while upper (resp. seed) boost is denoted by $\mathscr{T}^{(\cdot)_u}(\alpha)$ (resp. $\mathscr{T}^{(\cdot)_s}(\alpha, X)$).

## 4. Experimental setup

We ran various experiments using the two collections Robust2004 and Web2014 provided in the context of the TREC evaluation campaigns as well as the LETOR 3.0 collection. The experiments aimed to evaluate various variants of our approach, i.e., varying several components, to answer several evaluation questions.

### 4.1. Questions

The following four evaluation questions **Q1–Q4** guided the various experiments presented in the remainder of the paper.

**Q1** How does the set of features, considered as document skills, impact ranking? (see Section 5.1)
**Q2** What is the impact of the different parameters in terms of effectiveness and efficiency? (see Section 5.2)

**Table 3**
Description of the 15 features used for experiments on the Web2014 and Robust2004 collections. $\mathscr{F}_{QD}$, $\mathscr{F}_{QI}$ and $\mathscr{F}_{M}$ respectively stand for *Query-Dependent*, *Query-Independent* and *Model* features.

| ID | Feature | Robust2004 | Web2014 | Type |
|---|---|---|---|---|
| $f_1$ | $\sum_{t_i \in q \cap d} TF(t_i, d)$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_2$ | $\sum_{t_i \in q \cap d} IDF(t_i)$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_3$ | $\sum_{t_i \in q \cap d} TF(t_i, d) \cdot IDF(t_i)$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_4$ | $\sum_{t_i \in q \cap d} \log(TF(t_i, d))$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_5$ | $LEN(d)$ | √ | √ | $\mathscr{F}_{QI}$ |
| $f_6$ | $\sum_{t_i \in q \cap d} \frac{TF(t_i, d)}{LEN(d)}$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_7$ | $\sum_{t_i \in q \cap d} \frac{\log((TF(t_i, d))}{LEN(d)}$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_8$ | $\sum_{t_i \in q \cap d} \log\left(\frac{\lvert C\rvert}{TF(t_i, C)} + 1\right)$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_9$ | $\sum_{t_i \in q \cap d} TF(t_i, d) \cdot (\log(\lvert C\rvert . IDF(t_i))$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_{10}$ | $\sum_{t_i \in q \cap d} \log\left(\frac{TF(t_i, d)}{LEN(d)} \cdot \frac{\lvert C\rvert}{TF(t_i, C)} + 1\right)$ | √ | √ | $\mathscr{F}_{QD}$ |
| $f_{11}$ | Tf.Idf (Indri implementation) | √ | √ | $\mathscr{F}_{M}$ |
| $f_{12}$ | Okapi-BM25 with $k1 = 1.2$, $b = 0.75$, and $k3 = 7$ (Indri implementation) | √ | √ | $\mathscr{F}_{M}$ |
| $f_{13}$ | Indri proper model | √ | √ | $\mathscr{F}_{M}$ |
| $f_{14}$ | PageRank | | √ | $\mathscr{F}_{QI}$ |
| $f_{15}$ | Spam score | | √ | $\mathscr{F}_{QI}$ |

**Q3** Which parameter combination yields effective ranking? (see Section 5.3)
**Q4** Do document competitions outperform data fusion methods or LTR ones ? (see Section 5.4)

### 4.2. Collections

We used 9 different datasets for experiments, with a total of 981 queries:

- The Robust2004 collection provided for the TREC 2004 Robust Retrieval Track. The corpus is composed of 528,155 homogeneous documents (disks 4 and 5 without Congressional Record—CR—documents). 249 topics are available (Voorhees, 2004).
- The Web2014 collection provided for the TREC 2014 Web Track. This collection is commonly considered as a large-scale collection of heterogeneous documents. It is composed of the ClueWeb12-full dataset,[3] which comprises 733 million pages (27.3 TB), and a total of 50 topics (Collins-Thompson, Macdonald, Bennett, Diaz, & Voorhees, 2014).
- The LETOR 3.0 collection provided by Microsoft Research for evaluating and comparing LTR algorithms.[4] It is composed of 7 datasets, one based on the Ohsumed collection and the others based on the ".gov" collection (HP2003, HP2004, NP2003, NP2004, TD2003, TD2004). Each dataset provides from 50 to 150 topics (a complete description is available in Liu, 2011).

### 4.3. Feature description

Depending on the collection, we used different sets of features. For Ohsumed and ".gov", we directly used the features from the LETOR 3.0 datasets (Liu, 2011) (Ohsumed provides 45 features versus 64 for the ".gov" collection). 13 features were defined for the Robust2004 collection. Features $f_1$ to $f_{10}$ are usual features used in LTR, taken from Liu (2011). Features $f_{11}$, $f_{12}$, and $f_{13}$ correspond to the state-of-the-art models implemented in Indri,[5] i.e., feature $f_{11}$ corresponds to Tf.Idf, feature $f_{12}$ corresponds to Okapi, and feature $f_{13}$ corresponds to the Indri model, which is a combination of language modeling and inference network retrieval frameworks (Metzler, 2011). These 13 features were also used for the Web2014 collection, complemented with two Web specific features (i.e., PageRank and spam scores[6]). According to their definitions, features are categorized into three classes: Query-Independent ($\mathscr{F}_{QI}$), Query-Dependent ($\mathscr{F}_{QD}$), and Model ($\mathscr{F}_{M}$). The features associated to the Robust2004 and Web2014 collections are summarized in Table 3. All the features fall into the case 1 described in Section 2.3.1, were normalized between 0 and 1 using a max normalization, and were evaluated on whole documents.

[3] http://www.lemurproject.org/clueweb12.php/.
[4] http://research.microsoft.com/en-us/um/beijing/projects/letor//letor3dataset.aspx.
[5] http://www.lemurproject.org/doxygen/lemur/html/IndriRunQuery.html.
[6] http://lemurproject.org/clueweb12/related-data.php.

**Table 4**
Parameter values used in experiments. Parameters in bold were used during the 5-fold cross-validation.

| Parameter | Values |
|---|---|
| $\mathscr{F}$ | $\mathscr{F}_{\mathbf{M}}$; $\mathscr{F}_{\mathbf{QD}}$; $\mathscr{F}_{QI}$; $\mathscr{F}_{\mathbf{M,QD}}$; $\mathscr{F}_{\mathbf{M,QI}}$; $\mathscr{F}_{QD,QI}$; $\mathscr{F}_{\mathbf{All}} = \mathscr{F}_{\mathbf{M,QD,QI}}$ |
| $\mathscr{T}$ | $\mathscr{T}^{\mathbf{RR}}$; |
| | $\mathscr{T}^{\mathbf{SS}}(\mathbf{r = 10})$; $\mathscr{T}^{\mathbf{SS}}(\mathbf{r = 20})$; $\mathscr{T}^{\mathbf{SS}}(\mathbf{r = 50})$; |
| | $\mathscr{T}^{\mathbf{PRR}}(\rho = \mathbf{2}, \mathbf{N = 10})$; $\mathscr{T}^{\mathbf{PRR}}(\rho = \mathbf{2}, \mathbf{N = 20})$; |
| | $\mathscr{T}^{PRR}(\rho = 5, 10)$; $\mathscr{T}^{\mathbf{PRR}}(\rho = \mathbf{5}, \mathbf{N = 20})$; |
| | $\mathscr{T}^{PSS}(r = 10, \rho = 2, N = 10)$; $\mathscr{T}^{PSS}(r = 10, \rho = 2, N = 20)$; |
| | $\mathscr{T}^{PSS}(r = 10, \rho = 5, N = 10)$; $\mathscr{T}^{PSS}(r = 10, \rho = 5, N = 20)$; |
| | $\mathscr{T}^{\mathbf{PSS}}(\mathbf{r = 20}, \rho = \mathbf{2}, \mathbf{N = 10})$; $\mathscr{T}^{\mathbf{PSS}}(\mathbf{r = 20}, \rho = \mathbf{2}, \mathbf{N = 20})$; |
| | $\mathscr{T}^{PSS}(r = 20, \rho = 5, N = 10)$; $\mathscr{T}^{PSS}(r = 20, \rho = 5, N = 20)$; |
| | $\mathscr{T}^{\mathbf{PSS}}(\mathbf{r = 50}, \rho = \mathbf{2}, \mathbf{N = 10})$; $\mathscr{T}^{PSS}(r = 50, \rho = 2, N = 20)$; |
| | $\mathscr{T}^{PSS}(r = 50, \rho = 5, N = 10)$; $\mathscr{T}^{\mathbf{PSS}}(\mathbf{r = 50}, \rho = \mathbf{5}, \mathbf{N = 20})$ |
| Boosting | $\mathscr{T}^{(\cdot)}$; |
| | $\mathscr{T}^{(\cdot)\mathbf{u}}(\boldsymbol{\alpha} = \mathbf{3})$; $\mathscr{T}^{(\cdot)\mathbf{u}}(\boldsymbol{\alpha} = \mathbf{5})$; |
| | $\mathscr{T}^{(\cdot)\mathbf{s}}(\boldsymbol{\alpha} = \mathbf{3}, \mathbf{X = 10})$; $\mathscr{T}^{(\cdot)\mathbf{s}}(\boldsymbol{\alpha} = \mathbf{3}, \mathbf{X = 20})$; |
| | $\mathscr{T}^{(\cdot)s}(\alpha = 5, X = 10)$; $\mathscr{T}^{(\cdot)\mathbf{s}}(\boldsymbol{\alpha} = \mathbf{5}, \mathbf{X = 20})$ |
| Impact | 1 ($\Lambda = 20\%$); 1 ($\Lambda = 50\%$); 1 ($\Lambda = \infty$); |
| | **distance** ($\Lambda = \mathbf{100}\%$); **distance** ($\Lambda = \mathbf{200}\%$); |
| | **distance** ($\Lambda = \infty$) |
| Strategy type | $\boldsymbol{\sigma^{\mathbf{v}}}$;$\boldsymbol{\sigma^{\mathbf{r}}}$ |

### 4.4. Protocol

The parameter values used to evaluate our model are described in Table 4. One can refer to Table 1 for a notation remainder. Concerning the impact, two main cases are evaluated: systematically decreasing the life gauge by 1 (denoted 1 in the table) or applying the $\varphi_i$ function, i.e., $\varphi_i$ is the normalized distance between two feature values (case 1 in Section 2.3.1). Several values of the initial life gauge $\Lambda$ are considered:

- when decreasing the gauge by 1, the initial gauge is set to 20% or 50% of the number of features (i.e., a document can be *dead* before all the features have been played), as well as $\infty$, i.e., all the matches are completely run using all the features.
- similarly, when decreasing $\Lambda$ using the $\varphi_i$ function, we set the initial gauge to 100%, 200% of the number of features, or $\infty$.[7] The applied percentage is $\geq$ 100 since the impact for each feature is in this case $\geq$ 1.

In all experiments *pw* (resp. *pd*) was set to 3 (resp. 1) following the rules of soccer leagues which aim to promote victories over draws.

All the runs were evaluated w.r.t. effectiveness computed using some usual measures (MAP, P@20, and MRR). We applied randomization tests to evaluate the statistical significance of the differences between run means as recommended in Smucker, Allan, and Carterette (2007).

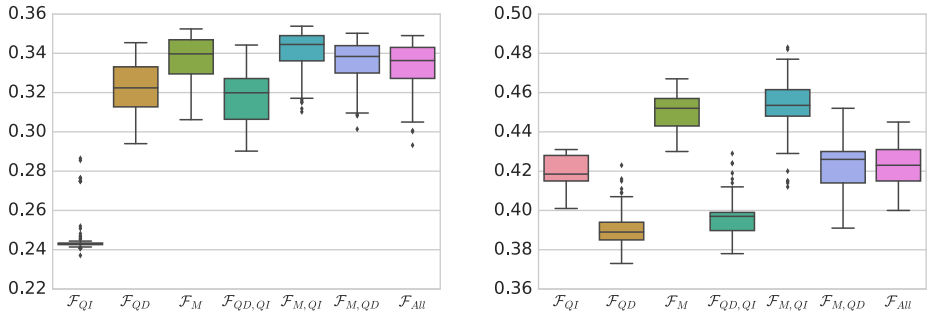
Concerning the Web2014 and Robust2004 collections, the qualified documents were the 50 top-ranked documents returned using the Indri model, which was the model retrieving the most relevant documents for both collections. Similarly, the best feature per dataset was used to select the top 50 documents to be qualified in LETOR 3.0. Some other experiments were also run on all documents (up to 1000 per query) of the dataset; however, as these results were less significant, they will not be detailed here.

To answer **Q1** we ran several experiments on the Web2014 and Robust2004 collections varying the set of features as described in the first line of Table 4. To fairly evaluate the impact of features, for each case (7 in total), 100 experiments were run randomly fixing all the other parameters among the values listed in Table 4 (lines 2–5).

The protocol used to answer **Q2** is very similar: for each parameter we aimed to evaluate the behavior, i.e., the tournament type, the boosting technique, the impact (when a document loses on a feature the way its life gauge is decreased), and the strategy type (*order-by-value, order-by-rank*), we run 100 experiments randomly fixing the other parameters.

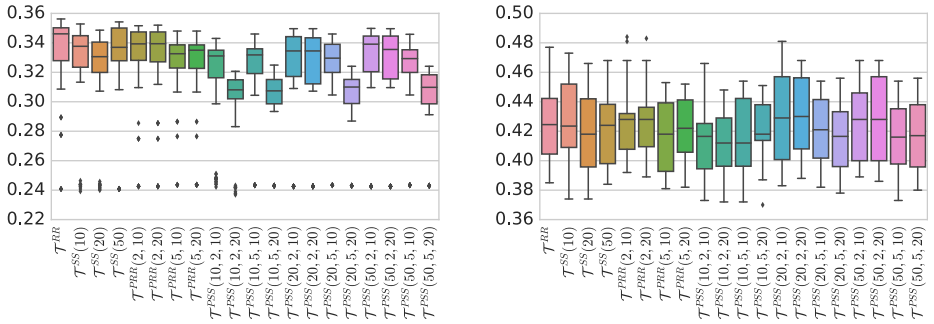
To answer **Q3** and **Q4**, three variants of TOURNARANK were compared to state-of-the-art approaches: two variants were chosen according to the results of **Q1** and **Q2**, and one variant was issued from a 5-fold cross validation. More details about this cross-

[7] A pilot study was conducted and showed that considering a life gauge $\geq$ 200% is similar to $\infty$. For clarity purpose we thus only provide results for these initial gauge values.




(a) Robust2004 (b) Web2014

**Fig. 5.** Results varying the set of features (P@20).




(a) Robust2004 (b) Web2014

**Fig. 6.** Results varying the tournament type (P@20).

validation methodology are given in Section 5.4. It should be recalled that contrary to LTR approaches, TOURNARANK is an unsupervised approach. The cross validation is here applied to select the best parameter combination to apply on the test dataset. These 3 variants were chosen to optimize the P@20 metric since this metric looks consistent to evaluate the quality of a reranking approach.

**Competitors.** Since TOURNARANK may be related to data fusion methods, we compared it with state-of-the-art rank and score aggregation methods (Borda (de Borda, 1781), RRF with k = 60 (Cormack, Clarke, & Buettcher, 2009), CombMNZ, and CombSum (Fox & Shaw, 1994)). We also compared our unsupervised approach to standard supervised ones, using 8 standard LTR methods (MART, RankNet, RankBoost, AdaRank, Coordinate Ascent, LambdaMART, ListNet, and Random Forests) that were executed on the 50 documents described previously.[8] These LTR methods were trained to optimize the P@20 metric to be fairly compared to TOURNARANK. A direct use of learning to rank methods included in LETOR 3.0 was not possible since the results are reported using up to 1000 documents per topic whereas we choose to qualify 50 documents.

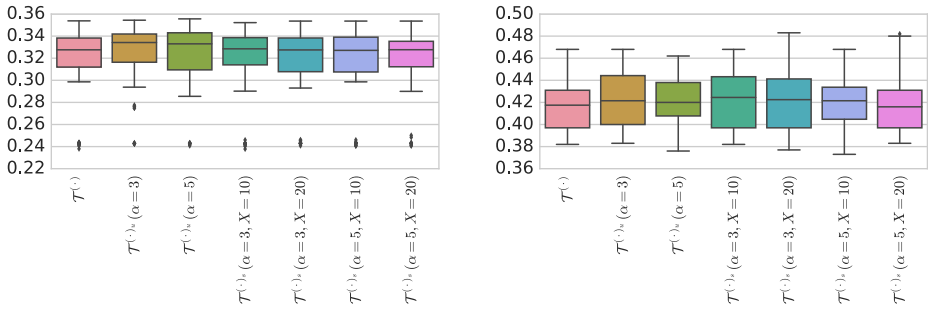
## 5. Results

The representative results are reported here and serve as basis to answer the aforementioned research questions.

### 5.1. Impact of the feature set

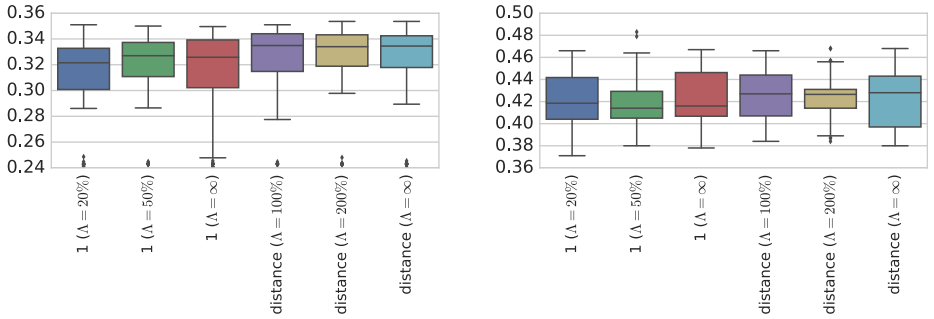
The results varying the feature set (see Fig. 5) provide answers to **Q1**. For brevity, the results are reported regarding the P@20 metric only. Similar conclusions can be drawn regarding the results over the other metrics, i.e., MAP and MRR. Generally, one can notice that the results are consistent on both collections. The only exception is when using the query-independent features only ($\mathcal{F}_{QI}$). Indeed, the results are poor on the Robust2004 dataset whereas they are satisfactory on the Web2014 dataset. This can be easily

[8] We used the RANKLIB library (https://sourceforge.net/p/lemur/wiki/RankLib/).



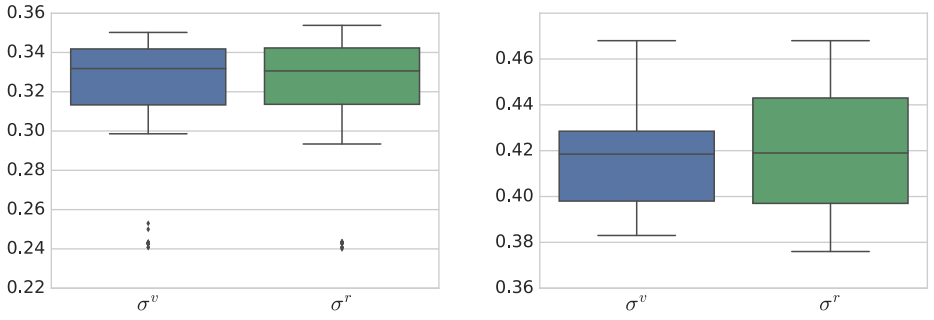

(a) Robust2004 (b) Web2014

**Fig. 7.** Results varying the boosting technique (P@20).




(a) Robust2004 (b) Web2014

**Fig. 8.** Results varying the impact (P@20).




(a) Robust2004 (b) Web2014

**Fig. 9.** Results varying the strategy type (P@20).

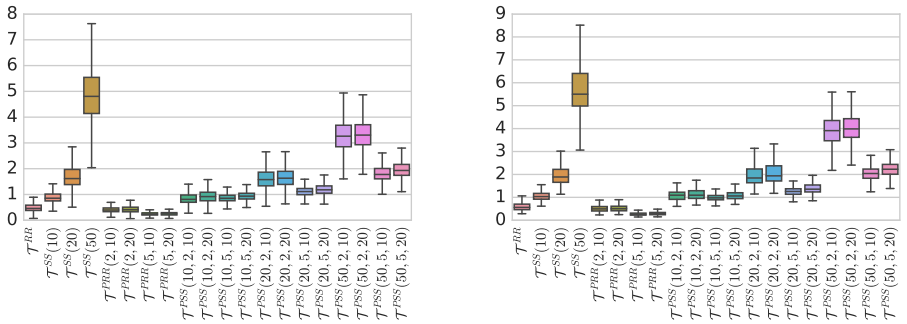



**Fig. 10.** Efficiency varying the tournament type (seconds).

**Table 5**
Comparison of 3 TOURNARANK variants with learning to rank and data fusion methods on the Web2014 and Robust2004 collections. For each collection, the fourth (resp. fifth) line presents results for the best (resp. worst) LTR method while the sixth line presents the results for the best data fusion method. † and ⋆ denote significance ($p < .05$) with respect to INDRI and the best data fusion method respectively. These symbols are doubled when $p < 0.01$. Randomization tests were not performed on the LTR methods.

| | Run | MAP | P@20 | MRR |
|---|---|---|---|---|
| Web2014 | TOURNARANK (median) | 0.1192$^{\dagger\dagger}$ | **0.4670**$^{\dagger\dagger}$ | 0.4786$^{\star}$ |
| | TOURNARANK (max) | 0.1190$^{\dagger\dagger, \star}$ | 0.4580† | 0.4921 |
| | TOURNARANK (cross) | 0.1176$^{\dagger, \star}$ | 0.4420 | 0.5585† |
| | RankBoost | **0.5091** | 0.4650 | 0.5400 |
| | ListNet | 0.4753 | 0.4110 | 0.4200 |
| | Borda | 0.1223$^{\dagger\dagger}$ | 0.4660$^{\dagger\dagger}$ | **0.5746**$^{\dagger\dagger}$ |
| | INDRI | 0.1132 | 0.4250 | 0.4299 |
| Robust2004 | TOURNARANK (median) | 0.1858† | 0.3494 | 0.6906 |
| | TOURNARANK (max) | 0.1864$^{\dagger\dagger, \star}$ | 0.3534 | 0.6893 |
| | TOURNARANK (cross) | 0.1798 | 0.3473 | 0.6769 |
| | CoordinateAscent | **0.4835** | **0.3639** | 0.5480 |
| | ListNet | 0.4605 | 0.3167 | 0.4800 |
| | RRF | 0.1832 | 0.3536 | **0.7000** |
| | INDRI | 0.1817 | 0.3490 | 0.6773 |

explained since the only query-independent feature on Robust2004 is the document length which is not sufficient to correctly rerank documents. Alternatively, in case of Web documents, more sophisticated features, i.e., PageRank and Spam score, tend to be reliable relevance clues. In both cases (Web2014 and Robust2004), the best performances are obtained when considering $\mathscr{F}_{M,QI}$, i.e., using model and query-independent features. This motivates the need for considering heterogeneous feature types. Indeed, as shown by the results of $\mathscr{F}_{M,QI}$ vs. $\mathscr{F}_{M,QD}$, the benefits of combining high-quality query-dependent features ($\mathscr{F}_{M}$) and query independent features ($\mathscr{F}_{QI}$) are higher than when combining features relying on the same concepts, i.e., TF and IDF ($\mathscr{F}_{M}$ and $\mathscr{F}_{QD}$). Finally, TOURNARANK is robust when considering the full set of features if it contains high-quality features, i.e., $\mathscr{F}_{M}$. Indeed, if we consider all the features ($\mathscr{F}_{All}$), the results slightly deteriorate. Consequently, the often costly feature selection step does not seem to be mandatory when running TOURNARANK over an unknown collection.

### 5.2. Impact of the TOURNARANK parameters

The results varying the TOURNARANK parameters (see Figs. 6–9) provide answers to **Q2** in terms of effectiveness.

**Tournament type.** In Fig. 6, we can see that effectiveness is quite similar whatever the considered tournament type on both

**Table 6**
Comparison of 3 TOURNARANK variants with learning to rank and data fusion methods on the LETOR collections. For each collection, the fourth (resp. fifth) line presents the results for the best (resp. worst) LTR method while the sixth line presents results for the best data fusion method. † and ⋆ denote significance ($p < .05$) with respect to the initial ranking and the best data fusion method respectively. These symbols are doubled when $p < .01$. Randomization tests were not performed on the LTR methods.

| | Run | MAP | P@20 | MRR |
|---|---|---|---|---|
| HP2003 | TOURNARANK (median) | 0.6449† | 0.0530 | 0.6859† |
| | TOURNARANK (max) | 0.6430† | 0.0533 | 0.6856† |
| | TOURNARANK (cross) | 0.6348†† ⋆ | 0.0530 | 0.6784† |
| | RankBoost | **0.7538** | **0.0550** | 0.6800 |
| | AdaRank | 0.2965 | 0.0417 | 0.1933 |
| | CombMNZ | 0.6634 | 0.0530 | 0.7029 |
| | Initial ranking | 0.7024 | 0.0537 | **0.7375** |
| HP2004 | TOURNARANK (median) | 0.5990 | 0.0507 | 0.6120 |
| | TOURNARANK (max) | 0.5988 | 0.0507 | 0.6121 |
| | TOURNARANK (cross) | 0.5514 | 0.0487 | 0.5636 |
| | RankBoost | **0.6783** | **0.0513** | 0.5733 |
| | ListNet | 0.1194 | 0.0227 | 0.0933 |
| | CombMNZ | 0.5547 | 0.0500 | 0.5673 |
| | Initial ranking | 0.6169 | **0.0513** | **0.6239** |
| NP2003 | TOURNARANK (median) | 0.6522†⋆ | 0.0457 | 0.6496†⋆ |
| | TOURNARANK (max) | 0.6522†⋆ | 0.0457 | **0.6522**†⋆ |
| | TOURNARANK (cross) | 0.6477†⋆ | 0.0453 | 0.6471†⋆ |
| | RankBoost | **0.6671** | **0.0473** | 0.5533 |
| | AdaRank | 0.2319 | 0.0313 | 0.1400 |
| | RRF | 0.5959 | 0.0457 | 0.5951 |
| | Initial ranking | 0.5777 | 0.0433 | 0.5792 |
| NP2004 | TOURNARANK (median) | 0.5901 | 0.0473 | 0.5954 |
| | TOURNARANK (max) | 0.5912 | 0.0473 | 0.5962 |
| | TOURNARANK (cross) | 0.5185 | **0.0480**† | 0.5245 |
| | Random Forests | 0.5974 | **0.0480** | 0.4800 |
| | ListNet | 0.1289 | 0.0313 | 0.0267 |
| | CombMNZ | **0.6204** | 0.0460† | **0.6193**† |
| | Initial ranking | 0.5197 | 0.0427 | 0.5186 |

collections. However, results are more variable on Robust2004. This is related to the observations discussed before about the impact of the feature set, i.e., a non negligible number of experiments where run using $\mathscr{F}_{QI}$ which leads to poor results. This observation holds for the rest of this section. We will discuss the default tournament type we recommend at the end of this section after having considered the efficiency aspect.

**Boosting technique.** In Fig. 7, we can see that the boosting technique does not significantly impact effectiveness. Consequently, running TOURNARANK without boosting seems to be a better choice since it does not imply the setting of extra parameters.

**Impact.** In Fig. 8, we can see on both collections that applying the $\varphi_i$ function leads to slightly better results than systematically decreasing the life gauge by 1. This shows the interest of an impact based on feature values rather than simply based on the order function. This motivates the investigation of subtle $\varphi_i$ functions, left as future work.

**Strategy type.** In Fig. 9, we can see that the strategy type does not significantly impact effectiveness. This may be due to the similarity in top positions between the two generated feature rankings. Whatever the initial life gauge of documents, the same features are played leading thus to similar results.

**Efficiency.** A counterintuitive observation can be done analyzing Fig. 10. The Swiss system tournaments are more time-consuming than the Round Robin ones. This is due to the pairing phase (line 5 of Algorithm 1) that counterbalances the gain regarding the number of matches. As expected (see Section 2.2.4), pooling documents leads to a better efficiency in all the cases. To conclude we suggest considering pooling Round Robin tournaments, which meets a good trade-off between efficiency and effectiveness.

**Table 7**
Comparison of 3 TOURNARANK variants with learning to rank and data fusion methods on the LETOR collections. For each collection, the fourth (resp. fifth) line presents the results for the best (resp. worst) LTR method while the sixth line presents results for the best data fusion method. † and ⋆ denote significance ($p < .05$) with respect to the initial ranking and the best data fusion method respectively. These symbols are doubled when $p < .01$. Randomization tests were not performed on the LTR methods.

| | Run | MAP | P@20 | MRR |
|---|---|---|---|---|
| TD2003 | TOURNARANK (median) | 0.1762 | 0.0980 | 0.4481 |
| | TOURNARANK (max) | 0.1760 | 0.1020 | 0.4505 |
| | TOURNARANK (cross) | 0.1634 | 0.0970 | 0.4376 |
| | RankBoost | **0.3929** | **0.1220** | 0.3400 |
| | ListNet | 0.2669 | 0.0770 | 0.1200 |
| | CombMNZ | 0.1757 | 0.1010 | 0.4444 |
| | Initial ranking | 0.1816 | 0.0970 | **0.4535** |
| TD2004 | TOURNARANK (median) | 0.1474⋆⋆ | 0.1613 | 0.4607 |
| | TOURNARANK (max) | 0.1479⋆⋆ | 0.1613 | 0.4650⋆ |
| | TOURNARANK (cross) | 0.1366 | 0.1587 | **0.4671**⋆ |
| | LambdaMART | **0.4193** | **0.1827** | 0.4267 |
| | ListNet | 0.2345 | 0.0913 | 0.1600 |
| | CombMNZ | 0.1309† | 0.1600 | 0.4010 |
| | Initial ranking | 0.1513 | 0.1620 | 0.4646 |
| OHSUMED | TOURNARANK (median) | 0.2523†† | 0.4349 | 0.6719† |
| | TOURNARANK (max) | 0.2525†† | 0.4363 | 0.6740 †, ⋆ |
| | TOURNARANK (cross) | 0.2574†, ⋆ | 0.4363 | 0.6738† |
| | RankBoost | **0.5363** | 0.4429 | 0.6226 |
| | MART | 0.4978 | 0.4165 | 0.5566 |
| | RRF | 0.2480†† | 0.4325 | 0.6370†† |
| | Initial ranking | 0.2675 | **0.4453** | **0.7342** |

### 5.3. Comparison of TOURNARANK configurations

In this section, we compare three different versions of TOURNARANK:

- TOURNARANK (median) corresponds to the parameters fixed according to the best median results in Figs. 6–9: $\mathscr{F}_{M,QI}$ (for Web2014 and Robust2004),[9] $\mathscr{T}^{PRR}(2, 20)$, $\mathscr{T}^{(\cdot)_S}(\alpha = 3, X = 10)$, distance ($\Lambda = \infty$), $\sigma^r$.
- TOURNARANK (max) corresponds to the parameters fixed according to the best max results in Figs. 6–9: $\mathscr{F}_{M,QI}$ (for Web2014 and Robust2004)[10], $\mathscr{T}^{RR}$, $\mathscr{T}^{(\cdot)_S}(\alpha = 3, X = 20)$, distance ($\Lambda = 200\%$), $\sigma^r$.
- TOURNARANK (cross) corresponds to a 5-fold cross-validation considering the 50% best parameter values regarding median results union the 50% best parameter values regarding max results. The best configuration obtained on the training set was launch on the test set for each fold. Those parameters are in bold in Table 4.

It should be noted that for Web2014 and Robust2014, data fusion methods were run on $\mathscr{F}_{M,QI}$ for fair comparison with TOURNARANK.

The results shown in Tables 5–7 provide answers to **Q3**.

The static configurations (max and median) obtain similar results w.r.t. all the evaluated metrics. Given that the max configuration slightly obtain better results, we advice to use this one as a preliminary choice on a new collection. Even better, this configuration performs better than when performing a cross-validation to learn collection-appropriate parameter values. Indeed TOURNARANK (max) yields better results than TOURNARANK (cross) on all the metrics on 4/9 datasets and on at least two metrics on almost all datasets (8/9). This shows that the max configuration is a robust configuration since it has been obtained from Robust2004 and Web2014 results while performing well on all the LETOR datasets.

### 5.4. Comparison with baselines and competitors

In this section, we compare the three aforementioned versions of TOURNARANK with the best and worst learning to rank competitors as well as with the best data fusion one over nine datasets. Results are summarized in Tables 5–7 and provide answers to **Q4**.

[9] For the other datasets, i.e., the LETOR3.0 collection, $\mathscr{F}_{All}$ was used for fair comparison w.r.t. LTR methods.

**Initial ranking.** A global remark is that TOURNARANK yields effective reranking w.r.t. at least one metric, i.e., MAP, P@20, or MRR, on most of the datasets (6/9). Moreover, on four datasets (Web2014, Robust2004, NP2003, and NP2004), all the three metrics are improved by TOURNARANK.

**Fusion methods.** With regard to the data fusion methods, TOURNARANK is most of the time more effective on the three metrics (5/9). When TOURNARANK fails in outperforming the best fusion method the performance is not statistically significant except in one case (Web2014). Moreover, on HP2003, the best fusion method failed in improving the initial ranking as well. In all the cases, TOURNARANK performs better than the worst data fusion method (not reported in Tables 5–7 since systematic). Finally, it should be noted that TOURNARANK systematically outperforms the best fusion methods using all the features on the Web2014 and Robust2004 collections. These results are not presented in the tables for clarity purpose.

**LTR techniques.** TOURNARANK systematically outperforms the worst LTR on MRR and P@20. Even better, TOURNARANK outperforms the worst LTR on all the metrics on 4/9 datasets (NP2003, NP2004, HP2003, and HP2004). Let us now compare TOURNARANK with the best LTR approaches. Even if they perform much better on the MAP metric, TOURNARANK systematically yields better results on the MRR and obtains comparable results according to P@20. It is interesting since LTR approaches have been trained to optimize the P@20 metric.

### 5.5. Discussion

Now we have carefully detailed our experiment results, we further analyze the implications of our results:

- Globally, TOURNARANK performs well on all the evaluated collection types. The results do not show any noticeable differences relative to the collection types (homogeneous or heterogeneous collections such as the Web2014 collection).
- The results of TOURNARANK are comparable to the best data fusion ones.
- TOURNARANK improves the initial ranking on P@20 and MRR, and is thus an effective reranking method on homogeneous and heterogeneous collections.
- Our best results on the P@20 and MRR metrics bear the idea that the top part of the ranking produced by TOURNARANK is of high quality. However, our more contrasted results on the MAP metric indicate that TOURNARANK has trouble identifying relevant documents in the soft underbelly of the initial list. A possible explanation of this phenomenon is that top-ranked documents won their matches thanks to some high-valued features. Moreover, when documents are represented by a majority of feature values close to the mean, TOURNARANK inherently experiences difficulties in distinguishing their relevance. This paves the way for more sophisticated document strategies.
- TOURNARANK is robust when considering the full set of features if it contains high-quality features. Indeed, if we consider all the features ($\mathcal{F}_{All}$), the results slightly deteriorate. Consequently, the often costly feature selection step does not seem to be mandatory when running TOURNARANK over an unknown collection.
- A recommended configuration has emerged from our results (max, see Section 5.3).
- This configuration have been obtained without any learning step and therefore can be used on any new datasets contrary to LTR approaches that require an available ground truth.

## 6. Related work

In this article we propose a novel approach to rank documents, by making them challenge each other, based on their feature values during matches. Since we consider a feature-based representation of documents and use some IR model scores as features, one may see similarities with the data fusion methods and the learning to rank ones. Furthermore, as only a subset of documents is qualified to participate to the tournament and as this qualification may depend on an initial ranking that may be used during matches, TOURNARANK can fall into the reranking category of approaches. In this section, we thus present the major document reranking approaches and then introduce the principles and major works in data fusion and learning to rank. Finally, the similarities and differences with our method are discussed.

### 6.1. Document reranking

Document reranking aims to reorder a list of documents retrieved by a search engine to improve the overall quality of this list, according to a given evaluation measure, e.g., P@$k$, MAP, MRR. In the literature, the reranking approaches lie in two categories (Ye, Huang, & Lin, 2011). The first category gathers *resource-based* approaches which use additional information or resources to rerank documents. These resources are for instance a thesaurus (Qu, Xu, & Wang, 2001), or in personalized search, user preferences (Daoud, Tamine-Lechani, Boughanem, & Chebaro, 2009; Fathy, Gharib, Badr, Mashat, & Abraham, 2014) or user behaviors (Cai, Liang, & de Rijke, 2014). Other methods, called graph-based, use inter-document relations as additional

information source, by building a graph of documents, using relations such as bibliographic relations (Picard & Savoy, 2000; Salton, Fox, & Voorhees, 1985) or, in a Web context, hyperlinks. Hits (Kleinberg, Kumar, Raghavan, Rajagopalan, & Tomkins, 1999) and PageRank (Page, Brin, Motwani, & Winograd, 1999) are the first state-of-the-art algorithms using such graphs. These approaches were generalized by Balinski and Danilowicz (2005) to allow the use of any inter-document relation with any retrieval model. The relationships are thus expressed by distances and can be obtained from the text, hyperlinks or other information. In Kurland and Lee (2005), a graph is recreated if hyperlinks are not available in the document collection. The second group of approaches *uses only documents as information source* and our method belongs to this category. These approaches use several kinds of document features. In Zhang, Chang, Zheng, Metzler, and Nie (2009), the document dates when queries are time-sensitive. In Ye et al. (2011), "richer" features are used, such as document length, average proximity of query terms, word features, or a Mitra equation (Mitra, Singhal, & Buckley, 1998). However, this latter approach is based on a learning method, where the documents used for the learning process are pseudo-relevant and non relevant documents. In Zheng and Cox (2009), each document is analyzed independently and a specificity score, based on the normalized inverse document frequency and the term entropies, is evaluated. In Lee, Park, and Choi (2001), documents are analyzed using document clusters, some of the clusters being tailored to the characteristics of the query.

### 6.2. Data fusion

Data fusion aims to combine multiple information resulting from different retrieval models or different query formulations, in order to provide the user with an improved and consensual list of results (Wu & Crestani, 2015). This list is obtained by aggregating either ranks or scores in an unsupervised way. Rank aggregation methods can be grouped into three categories: distributional-based, heuristic, and stochastic optimization algorithms (Lin, 2010). In IR, the most widely used are heuristic algorithms. They mainly aim to aggregate a small number of long lists. Among the most popular in IR, one can cite Borda (de Borda, 1781), Condorcet (de Condorcet, 1785), RRF (Cormack et al., 2009), or Markov chain methods (Dwork, Kumar, Naor, & Sivakumar, 2001). The relevance score aggregation methods require a scoring function for each ranked list and then combine, for each document, its scores and rank the document according to this aggregated score. The most known algorithms in IR are CombSum, whose the aggregated score is the sum of the document scores on ranked lists, and combMNZ, whose aggregated score is based on combSum and takes into account the number of systems that retrieved the document (Fox & Shaw, 1994).

### 6.3. Learning to rank

Learning to rank (LTR) aims to automatically optimize a ranking function using a machine learning algorithm and learning data, in order to rank documents or web pages according to their relevance for a given query (Li, 2014; Liu, 2011). Learning data consist of query-document pairs represented as feature vectors and associated to relevance judgments given as ground truth. Standard features are similarity scores obtained by IR models such as BM25 or language models. As a supervised learning task, LTR is composed of a training phase and a testing phase. In the training phase, the LTR algorithms use learning data to learn a ranking function that provides an optimized list of results. In the testing phase, the ranking function is used to predict the relevance of documents for unseen queries. A ranked list is thus obtained and is compared to the ground truth by using an evaluation measure. Several approaches have been proposed to solve the LTR task, among them are the regression-based algorithms (Cossock & Zhang, 2006), the classification-based algorithms (Nallapati, 2004), the pairwise classification-based algorithms such as RankNet (Burges et al., 2005), RankBoost (Freund, Iyer, Schapire, & Singer, 2003), and RankSVM (Chapelle & Keerthi, 2010; Joachims, 2002), as well as the algorithms that directly optimize IR evaluation measures, e.g., AdaRank (Xu & Li, 2007), or permutation counts in the ground truth list, e.g., ListNet (Cao, Qin, Liu, Tsai, & Li, 2007).

### 6.4. Discussion

Contrary to the data fusion methods that aggregate various ranked lists based on either the ranks or the scores of documents, our approach considers a single list of documents. TOURNARANK could nevertheless be seen as a fusion method when considering each feature as a list generator. One can point out some similarities between our approach and LTR. Indeed, both approaches represent documents in a feature space, using similar features. Moreover, a match between two documents can be seen as a pairwise comparison of documents based on their feature values, which might remind the reader with pairwise LTR. However, the major difference is that our approach is unsupervised, whereas LTR is a supervised learning task. Contrary to LTR, our approach does not require a training phase to learn a ranking function. Finally, with some configurations, TOURNARANK is clearly a

document reranking method.

## 7. Conclusion

### 7.1. Summary

In this article, we introduced TOURNARANK, a new and original unsupervised approach for document ranking which is not dedicated to a specific collection or domain. The documents are represented as feature sets which are then used by the documents to fight each other during matches organized in tournaments. The high flexibility and genericity of the model enables its adaptation to the reranking task. Our results on three state-of-the-art collections, i.e., TREC Robust2004, TREC Web2014, and LETOR 3.0, are very encouraging and enable investigating promising research lines. Before highlighting them, we now discuss some implications of our research.

### 7.2. Implications

From a *practical point of view*, the previously presented results show that a pre-processing step for feature selection is not vital to make TOURNARANK effective. Second, TOURNARANK does not require any labeled data since it is fully unsupervised. It consequently does not suffer from the well-known cold start problem. Third, the model is highly parallelizable since matches can be played simultaneously during *Round Robin* tournaments and per match day in *Swiss* tournaments.

From a *theoretical point of view*, TOURNARANK being a generic ranking model, it can therefore be applied on other IR domains from the moment documents can be described through features. For instance, TOURNARANK can be applied to image retrieval since images can be described by a variety of features (see Deselaers, Keysers, and Ney, 2008 for details). The match and tournament principles proposed in this article still hold for such data types. Another aspect of the TOURNARANK model is that it can be adapted to the aggregated search problem in the sense that different types of documents, e.g., images, videos, tweets, and web pages, can confront themselves. This idea is part of our future work and is thus further introduced in the next section.

### 7.3. Future work

Two directions are considered: model improvement and its evolution to the aggregated search issue. Regarding model improvement, TOURNARANK is totally new, highly modular, and some intuitive instances of the general framework presented here, i.e., the tournament type, the match rule, the feature set, and the document strategy, were studied in this article. However other possibilities remain uninvestigated. We first aim to evaluate other document features, that should be more heterogeneous and complementary than the ones presented here. Among them one may cite features related to credibility (Weerkamp & Rijke, 2012), readability (Polajnar, Glassey, & Azzopardi, 2012), or the information quantity delivered by documents (Shannon entropy (Shannon, 2001)). Second, alternative tournament types, e.g., introducing multi-stage tournaments, or match rules, e.g., where documents are not playing in turn, can be considered. Finally, we aim at enabling documents to learn their own strategies from past strikes/matches and to dynamically adapt their strategy during a match or a tournament.

For now TOURNARANK is only suited for ranking textual documents. The aggregated search issue has recently emerged within the IR community. Aggregated search attempt to achieve diversity by presenting search results from different information sources, e.g., images, blogs, or videos (Arguello, 2017; Kopliku, Pinel-Sauvagnat, & Boughanem, 2014; Lalmas, 2011). The challenge is to identify the subset of heterogeneous documents that best fits the user's need. To tackle this challenge, the TOURNARANK model can consider tournament between teams of heterogeneous documents. It raises some open questions related to team composition, e.g., introducing diversity or complementary criteria in team building, as well as the redefinition of the match concept when teams instead of independent documents fight themselves.

## References

Arguello, J. (2017). Aggregated search. *Foundations and Trends® in Information Retrieval, 10*(5), 365–502.

Balinski, J., & Danilowicz, C. (2005). Re-ranking method based on inter-document distances. *Information Processing and Management, 41*(4), 759–775. http://dx.doi.org/10.1016/j.ipm.2004.01.006.

de Borda, J.-C. (1781). *Mémoire sur les élections au scrutin. Histoire de l'académie royale des sciences. Paris*.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., et al. (2005). *Learning to rank using gradient descent. Proceedings of the 22nd international conference on machine learningICML'05*New York, NY, USA: ACM89–96. http://dx.doi.org/10.1145/1102351.1102363.

Cai, F., Liang, S., & de Rijke, M. (2014). *Personalized document re-ranking based on bayesian probabilistic matrix factorization. Proceedings of the 37th international acm sigir conference on research and development in information retrievalSIGIR'14*New York, NY, USA: ACM835–838. http://dx.doi.org/10.1145/2600428.2609453.

Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F., & Li, H. (2007). *Learning to rank: From pairwise approach to listwise approach. Proceedings of the 24th international conference on*

*machine learningICML'07*New York, NY, USA: ACM129–136. http://dx.doi.org/10.1145/1273496.1273513.
Chapelle, O., & Keerthi, S. S. (2010). Efficient algorithms for ranking with svms. *Information Retrieval, 13*(3), 201–215. http://dx.doi.org/10.1007/s10791-009-9109-9.
Collins-Thompson, K., Macdonald, C., Bennett, P. N., Diaz, F., & Voorhees, E. M. (2014). *TREC 2014 Web Track Overview. Proceedings of the twenty-third text retrieval conferenceTREC'14*.
de Condorcet, N. (1785). *Essai sur l'application de l'analyse ã la probabilité des décisions rendues à la pluralité des voix.* Paris: Imprimerie Royale.
Cormack, G. V., Clarke, C. L. A., & Buettcher, S. (2009). *Reciprocal rank fusion outperforms condorcet and individual rank learning methods. Proceedings of the 32nd international acm sigir conference on research and development in information retrievalSIGIR'09*New York, NY, USA: ACM758–759. http://dx.doi.org/10.1145/1571941.1572114.
Cossock, D., & Zhang, T. (2006). *Subset ranking using regression. Proceedings of the 19th annual conference on learning theoryCOLT'06*Berlin, Heidelberg: Springer-Verlag605–619. http://dx.doi.org/10.1007/11776420_44.
Daoud, M., Tamine-Lechani, L., Boughanem, M., & Chebaro, B. (2009). *A session based personalized search using an ontological user profile. Proceedings of the 2009 acm symposium on applied computingSAC'09*New York, NY, USA: ACM1732–1736. http://dx.doi.org/10.1145/1529282.1529670.
Deselaers, T., Keysers, D., & Ney, H. (2008). Features for image retrieval: An experimental comparison. *Information Retrieval, 11*(2), 77–107. http://dx.doi.org/10.1007/s10791-007-9039-3.
Dwork, C., Kumar, R., Naor, M., & Sivakumar, D. (2001). *Rank aggregation methods for the web. Proceedings of the 10th international conference on world wide webWWW'01*New York, NY, USA: ACM613–622. http://dx.doi.org/10.1145/371920.372165.
Edmonds, J. (1965). Paths, trees, and flowers. *Canadian Journal of mathematics, 17*(3), 449–467. http://dx.doi.org/10.4153/CJM-1965-045-4.
Fathy, N., Gharib, T. F., Badr, N., Mashat, A. S., & Abraham, A. (2014). A personalized approach for re-ranking search results using user preferences. *Journal of Universal Computer Science, 20*(9), 1232–1258.
Fox, E. A., & Shaw, J. A. (1994). *Combination of multiple searches. The second text retrieval conferenceTREC-2500-215. The second text retrieval conference* NIST243–252.
Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research, 4*, 933–969.
Joachims, T. (2002). *Optimizing search engines using clickthrough data. Proceedings of the eighth acm sigkdd international conference on knowledge discovery and data miningKDD'02*New York, NY, USA: ACM133–142. http://dx.doi.org/10.1145/775047.775067.
Kleinberg, J. M., Kumar, R., Raghavan, P., Rajagopalan, S., & Tomkins, A. S. (1999). *The web as a graph: Measurements, models, and methods. Proceedings of the 5th annual international conference on computing and combinatoricsCOCOON'99*Berlin, Heidelberg: Springer-Verlag1–17. http://dl.acm.org/citation.cfm?id=1765751.1765753.
Kopliku, A., Pinel-Sauvagnat, K., & Boughanem, M. (2014). Aggregated search: A new information retrieval paradigm. *ACM Computing Surveys, 46*(3), 41:1–41:31. http://dx.doi.org/10.1145/2523817.
Kurland, O., & Lee, L. (2005). *Pagerank without hyperlinks: Structural re-ranking using links induced by language models. Proceedings of the 28th annual international acm sigir conference on research and development in information retrievalSIGIR'05*New York, NY, USA: ACM306–313. http://dx.doi.org/10.1145/1076034.1076087.
Lalmas, M. (2011). Aggregated search. In M. Melucci, & R. Baeza-Yates (Eds.). *Advanced topics in information retrieval* (pp. 109–123). Berlin, Heidelberg: Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-642-20946-8_5.
Lee, K.-S., Park, Y.-C., & Choi, K.-S. (2001). Re-ranking model based on document clusters. *Information Processing and Management, 37*(1), 1–14. http://dx.doi.org/10.1016/S0306-4573(00)00017-0.
Li, H. (2014). *Learning to rank for information retrieval and natural language processing* (2nd ed.). *Synthesis lectures on human language technologies*Morgan & Claypool Publishershttp://dx.doi.org/10.2200/S00607ED2V01Y201410HLT026.
Lin, S. (2010). Rank aggregation methods. *Wiley Interdisciplinary Reviews: Computational Statistics, 2*(5), 555–570. http://dx.doi.org/10.1002/wics.111.
Liu, J., & Belkin, N. J. (2015). Personalizing information retrieval for multi-session tasks: Examining the roles of task stage, task type, and topic knowledge on the interpretation of dwell time as an indicator of document usefulness. *Journal of the Association for Information Science and Technology, 66*(1), 58–81. http://dx.doi.org/10.1002/asi.23160.
Liu, T.-Y. (2011). *Learning to rank for information retrieval.* Springerhttp://dx.doi.org/10.1007/978-3-642-14267-3.
Macdonald, C., Santos, R. L., & Ounis, I. (2013). The whens and hows of learning to rank for web search. *Information Retrieval, 16*(5), 584–628. http://dx.doi.org/10.1007/s10791-012-9209-9.
Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval.* New York, NY, USA: Cambridge University Press.
Metzler, D. (2011). *A feature-centric view of information retrieval. The information retrieval series*, *27*Springerhttp://dx.doi.org/10.1007/978-3-642-22898-8.
Mitra, M., Singhal, A., & Buckley, C. (1998). *Improving automatic query expansion. Proceedings of the 21st annual international acm sigir conference on research and development in information retrievalSIGIR'98*New York, NY, USA: ACM206–214. http://dx.doi.org/10.1145/290941.290995.
Nallapati, R. (2004). *Discriminative models for information retrieval. Proceedings of the 27th annual international acm sigir conference on research and development in information retrievalSIGIR'04*New York, NY, USA: ACM64–71. http://dx.doi.org/10.1145/1008992.1009006.
Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The pagerank citation ranking: Bringing order to the webTechnical Report 1999-66.* Stanford InfoLab Previous number = SIDL-WP-1999-0120.
Picard, J., & Savoy, J. (2000). A logical information retrieval model based on a combination of propositional logic and probability theory. In F. Crestani, & G. Pasi (Eds.). *Soft computing in information retrieval: Techniques and applications* (pp. 225–258). Heidelberg: Physica-Verlag HD. http://dx.doi.org/10.1007/978-3-7908-1849-9_10.
Polajnar, T., Glassey, R., & Azzopardi, L. (2012). *Detection of news feeds items appropriate for children. Proceedings of the 34th european conference on advances in information retrievalECIR'12*Berlin, Heidelberg: Springer-Verlag63–72. http://dx.doi.org/10.1007/978-3-642-28997-2_6.
Qu, Y., Xu, G., & Wang, J. (2001). *Rerank method based on individual thesaurus. Proceedings of the third second workshop meeting on evaluation of chinese & japanese text retrieval and text summarizationNTCIR-2*.
Salton, G., Fox, E. A., & Voorhees, E. (1985). Advanced feedback methods in information retrieval. *Journal of the American Society for Information Science and Technology, 36*(3), 200–210. http://dx.doi.org/10.1002/asi.4630360311.
Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review, 5*(1), 3–55.
Smucker, M. D., Allan, J., & Carterette, B. (2007). *A comparison of statistical significance tests for information retrieval evaluation. Proceedings of the sixteenth acm conference on conference on information and knowledge managementCIKM'07*New York, NY, USA: ACM623–632. http://dx.doi.org/10.1145/1321440.1321528.
Tax, N., Bockting, S., & Hiemstra, D. (2015). A cross-benchmark comparison of 87 learning to rank methods. *Information Processing and Management, 51*(6), 757–772. http://dx.doi.org/10.1016/j.ipm.2015.07.002.
Voorhees, E. M. (2004). *Overview of the TREC 2004 Robust Track. Proceedings of the thirteenth text retrieval conferenceTREC'04*.
Weerkamp, W., & Rijke, M. (2012). Credibility-inspired ranking for blog post retrieval. *Inf. Retr. 15*(3–4), 243–277. http://dx.doi.org/10.1007/s10791-011-9182-8.
Wu, S., & Crestani, F. (2015). A geometric framework for data fusion in information retrieval. *Information Systems, 50*, 20–35. http://dx.doi.org/10.1016/j.is.2015.01.001.
Xu, J., & Li, H. (2007). *Adarank: A boosting algorithm for information retrieval. Proceedings of the 30th annual international acm sigir conference on research and development in information retrievalSIGIR'07*New York, NY, USA: ACM391–398. http://dx.doi.org/10.1145/1277741.1277809.
Ye, Z., Huang, J. X., & Lin, H. (2011). Incorporating rich features to boost information retrieval performance: A svm-regression based re-ranking approach. *Expert Systems with Applications, 38*(6), 7569–7574. http://dx.doi.org/10.1016/j.eswa.2010.12.108.

Zhai, C. (2015). *Towards a game-theoretic framework for information retrieval. Proceedings of the 38th international acm sigir conference on research and development in information retrievalSIGIR'15*New York, NY, USA: ACM 543–543. doi:10.1145/2766462.2767853.

Zhang, R., Chang, Y., Zheng, Z., Metzler, D., & Nie, J.-y. (2009). *Search result re-ranking by feedback control adjustment for time-sensitive query. Proceedings of human language technologies: The 2009 annual conference of the north american chapter of the association for computational linguistics, companion volume: Short papersNAACL-Short'09*Stroudsburg, PA, USA: Association for Computational Linguistics165–168.

Zheng, L., & Cox, I. J. (2009). *Re-ranking documents based on query-independent document specificity. Proceedings of the 8th international conference on flexible query answering systemsFQAS '09*Berlin, Heidelberg: Springer-Verlag201–214. http://dx.doi.org/10.1007/978-3-642-04957-6_18.