An Evaluation of Diversification Techniques

Duong Chi Thang, Nguyen Thanh Tam,
Nguyen Quoc Viet Hung(✉), and Karl Aberer

École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
{thang.duong,tam.nguyenthanh,quocviethung.nguyen,karl.aberer}@epfl.ch

Abstract. Diversification is a method of improving user satisfaction by increasing the variety of information shown to user. Due to the lack of a precise definition of information variety, many diversification techniques have been proposed. These techniques, however, have been rarely compared and analyzed under the same setting, rendering a 'right' choice for a particular application very difficult. Addressing this problem, this paper presents a benchmark that offers a comprehensive empirical study on the performance comparison of diversification. Specifically, we integrate several state-of-the-art diversification algorithms in a comparable manner, and measure distinct characteristics of these algorithms with various settings. We then provide in-depth analysis of the benchmark results, obtained by using both real data and synthetic data. We believe that the findings from the benchmark will serve as a practical guideline for potential applications.

1 Introduction

The diversification problem has been long acknowledged in information retrieval [8,10]. Differently from traditional information retrieval techniques which focus on the relevance of search results, diversification is a method of spreading a variety of data shown to user [14,23]. Improving diversity would increase not only the amount of information displayed with a limited number of data items but also the probability of delivering at least one piece of information that truly matches user intent. Moreover, focusing only on relevance might lead to redundancy and biases in the retrieval results due to similarity in structure and content of data items and coverage limitation of search engines [21]. For these reasons, diversity has become a crucial property to enhance user satisfaction through providing a general view that covers different aspects (i.e. subtopics) of data. Moreover, diversification is not only applicable to information retrieval but also numerous other domains, including web search [2,18], large-scale visualization [19], recommender systems [17,28], and novelty detection [12,22].

Diversification implies a trade-off between selecting data of relevance to user intent and filtering data having similar characteristics. As such, diversification is often characterized as a bi-criteria optimization problem, in which the twin objectives of being relevant and being dissimilar compete with each other [9]. To tackle this problem, a rich body of research has proposed different diversification

Q. Chen et al. (Eds.): DEXA 2015, Part II, LNCS 9262, pp. 215–231, 2015.
DOI: 10.1007/978-3-319-22852-5_19

techniques, ranging from *threshold-based* approach to *function-based* approach and *graph-based* approach. In general, the threshold-based approach defines a threshold on one criterion (i.e. either relevance or diversity), and then selects the data that both satisfy this threshold and optimize the other criterion. The function-based approach combines both relevance and diversity in a unified function, and then finds a set of data that maximizes this function. The graph-based approach models data items and their relationships as a graph, and then ranks the data according to the collective information inferred from the graph.

While many diversification techniques have been developed over the last decades, there has been little work on the evaluation of their performance altogether. To this end, various IR test collections have been created for diversity track such as TREC [7], NTCIR [16], and CLEF [4]. In that, participants are able to test their proposed methods using common real-world datasets and metrics [8]. Although these real-world datasets provide a pragmatic view, they do not allow participants to compare their methods in different settings such as the number of subtopics in input data or the number of displayed items. As a result, an algorithm may perform poorly within the common settings but may be able to achieve good performance under another different setting. In addition, the proposed methods may not be evaluated in a comparable manner as the evaluation is not conducted by a third party and under the same system. To this end, we present an evaluation of diversification techniques within a common benchmarking framework that offers the following salient features:

- We selected and implemented the most representative diversification techniques, including threshold-based approach: Swap [24] and Motley [15]; function-based approach: MMR [5] and MSD [13]; and graph-based approach: Affinity Graph [26] and GrassHopper [27]. In addition to comparison purposes, our framework provides reusable components to reduce the development time.
- We designed a generic, extensible benchmarking framework to assist in the evaluation of different diversification techniques, so that subsequent studies are able to easily compare their proposals with the state-of-the-art techniques.
- We compare the above diversification techniques in a fair manner using the same system and settings. Moreover, our experimental results are reliable, reproducible and extensible as the source code of the benchmarking framework as well as the datasets are publicly available.
- We simulated different settings of relevance and diversity measures. In particular, the framework allows users to vary configurable parameters and visualize their effects. Through empirical observations, user is given more insights and better understandings of the behavior of evaluated techniques.
- We offer extensive as well as intensive performance analyses. We believe that these analyses can serve as a practical guideline for how to select a well-suited diversification technique on particular application scenarios.

The remainder of this paper is organized as follows. Section 2 reviews state-of-the-art diversification techniques. We then discuss the benchmarking methodology in Sect. 3. Section 4 describes the experiments on both real and synthetic

data. Section 5 summarizes and concludes the benchmarking study, where we provide important suggestions for applications that need diversification.

2 Diversification Techniques

The problem of diversification can be formulated as follows. It takes as input a triple $\langle D, R, M \rangle$. In that, $D = \{d_1, \ldots, d_n\}$ is a set of data items. $R = \{r_1, \ldots, r_n\}$ is a set of relevance scores, in which each r_i is associated with each item d_i. $M = [m_{ij}]_{n \times n}$ is a matrix in which each element m_{ij} measures the dissimilarity between item d_i and item d_j. Given a limited-budget number of displayed items k ($k \ll n$), the problem output is a subset of items $D_k^* \subseteq D$ such that $|D_k^*| = k$ and the selected items are simultaneously having high relevance scores and being highly dissimilar among themselves. When the limited-budget number of displayed items k is clear from the context, we denote the result set D_k^* as D^*. In this section, we offer a description of the six diversification techniques studied in the benchmark. These diversification techniques are carefully selected based on the following criteria: (1) they are the representatives for their respective approach and (2) they are often referred in the research community.

Swap. Swap [24] is a *threshold-based* algorithm that firstly focuses on relevance and then gradually improves diversity. Technically, it initializes the output D^* with the top-k most relevant items and iteratively traverses the remaining items $D \setminus D^*$ in the decreasing order of relevance scores. In each iteration, the currently traversed item is swapped with the item in D^* that is the least dissimilar to the others in D^* if the diversity increases and the relevance drop is not below a pre-defined threshold. The process stops when there is no remaining item left to traverse.

Motley. Motley [15] is also a *threshold-based* algorithm. Unlike Swap, it starts from an empty set and constructs the output by incrementally adding data items in the decreasing order of relevance scores. An item d_i in turn is added to the output D^* if for every item $d_j \in D^*$, the dissimilarity m_{ij} is higher than a predefined threshold. The procedure stops when all items are considered or k items are already included to the output. In the case where all items are considered but there is still available budget, $k - |D^*|$ items from $D \setminus D^*$ are selected randomly and added to D^*.

Maximal Marginal Relevance (MMR). This algorithm belongs to the *function-based* approach in which the twin aspects of relevance and diversity are combined in a comprehensive objective function. More precisely, MMR [5] defines this objective function as $f(D^*) = (1-\lambda) \sum_{d_i \in D^*} r_i + 2\lambda \sum_{d_i, d_j \in D^*} m_{ij}$, where λ is a tunable parameter that specifies the preference between relevance and diversity. Since maximizing this function is NP-hard, MMR takes a greedy method that builds the output D^* incrementally with k iterations, in each of which an item d_i is selected if the value $\phi(d_i) = (1 - \lambda) r_i + \lambda \sum_{d_j \in D^*} m_{ij}$ is maximal. The core idea is that the objective function can be rewritten as

$f(D^*) = \sum_{d_i \in D^*} \phi(d_i)$ and maximizing the subterm $\phi(d_i)$ in each iteration is expected to approximately maximizing the whole function.

Max-Sum Dispersion (MSD). This is a *function-based* algorithm [13] like MMR that also takes a greedy method to maximize the above objective function. However, MSD takes another rewritten form: $f(D^*) = \sum_{d_i, d_j \in D^*} \varphi(d_i, d_j)$, where $\varphi(d_i, d_j) = \frac{1}{2}(1 - \lambda)(r_i + r_j) + 2\lambda m_{ij}$. The difference between MSD and MMR is that instead of selecting one item at a time, MSD picks two items d_i and d_j that have maximal $\varphi(d_i, d_j)$ value. When k is odd, the last item is selected randomly to add to the result set.

GrassHopper. This technique [27] belongs to the *graph-based* approach. It constructs a graph, which models the diversity and relevance of data items, from the dissimilarity matrix M and the relevance scores R. The graph serves as the representation of underlying states and transitions of an absorbing Markov chain which is used to rank the data items. Based on the Markov chain, the algorithm iterates between two routines: (1) turn the currently selected data items into absorbing states, and (2) add the new item with the highest expected number of visits before absorption to the output. As a consequence, the item set is diversified since the higher the expected number of visits, the more dissimilar between the new item and the previously selected items.

Affinity Graph (AG). AG [26] is also a *graph-based* algorithm, which ranks each data item by taking into account not only its relevance and its diversity but also its importance. The importance of an item is computed from the stationary distribution of a Markov chain whose transition matrix is constructed from the dissimilarity matrix M. AG in turn combines the relevance, the diversity, and the importance into a unified ranking score. The output D^* is then constructed by selecting the top-k items with the highest scores.

Summary: we have implemented representative algorithms in each approach. The implemented algorithms are AG, GrassHopper, MSD, MMR, Swap and Motley. Each algorithm exhibits various diversification characteristics. In fact, often these characteristics are not exclusive; a technique might have multiple ones. Table 1 features each implemented technique with the following key characteristics.

- **Balance:** algorithms that provide a tunable parameter to balance between diversity and relevance.
- **Data processing:** the ability to perform (online or offline) in response to the new arrival of data items. An online technique can process item-by-item in a serial fashion, whereas offline ones have to re-compute the whole result set.
- **#Parameters:** this characteristic is important as the higher number of parameters an algorithm requires, the harder for the users to configure the algorithm correctly.

It is noteworthy that all the diversification techniques require some parameters and the implementation of these techniques requires searching for a suitable

parameter settings, which is a critical issue to evaluate the techniques. In addition to the tradeoff parameter λ, some of these methods need other tunable parameters such as the damping factor in graph-based approach which might significantly affect the performance of the algorithms. However, finding the best parameters for each technique would be difficult; and even if we could find them, it would be unfair to the techniques with fewer parameters. Therefore, for fair comparison, all the parameter settings are decided based on the original authors' recommendation and fixed for all runs. In addition, although the pseudo code for most of the algorithms are available, there are many implementation details that we have to decide. For each decision, we strive for the option which would be fair for all algorithms and report the decision for future references on our website[1].

3 Benchmark Methodology

This section describes the setup used in the benchmark. We first present the system architecture of our benchmarking framework. Then, we provide the detail of two real-world datasets used in the benchmark. Next, we describe the methodology and procedure to generate synthetic data used in the experiments. Furthermore, we offer the descriptions of the measures used to assess the diversification techniques. Finally, we discuss the functionality of our benchmarking tool in analyzing the diversification results.

3.1 Framework

A primary goal of this study is to provide a flexible yet powerful tool to support the comparison and analysis of diversification techniques. To this end, we have developed a framework that employs the original performance study of these techniques. Figure 1 illustrates the component-based architecture of the framework, which is built upon three layers: *data access layer, computing layer*, and *application layer*. The data access layer abstracts the underlying data items (synthetic or real data) and feeds them to upper layers. The application layer interacts with users to receive configurable parameters and visualize outputs from the computing layer. The computing layer consists of two main components: diversification module and simulation module. While the former is plugged with diversification techniques that take data from the data access layer and deliver evaluation measures to the application layer, the latter is responsible for generating synthetic data that is designed to show how well the techniques perform in general.

We believe that subsequent studies are able to easily compare their proposals with the state-of-the-art techniques by using our framework. As presented, it is flexible and extensible, since a new technique as well as a new measurement can be easily plugged in. The framework is available for download from our website[2].

[1] https://code.google.com/p/diversity-benchmark.
[2] https://code.google.com/p/diversity-benchmark.

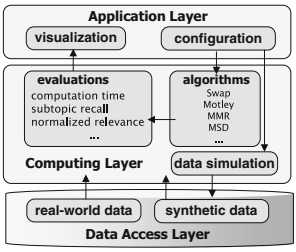

Fig. 1. Benchmarking Framework

Table 1. Characteristics of the algorithm

	Balance	Data processing	#Parameters
Swap	No	Offline	1
Motley	No	Online	1
MMR	Yes	Online	1
MSD	Yes	Online	1
AG	Yes	Offline	3
GrassHopper	No	Offline	1

3.2 Datasets

In this section, we show how to create the data input for the benchmark, which can adapt to both real-world data and synthetic data. While the former provides a pragmatic view, the latter offers different settings for deep examination of the algorithms.

Real-World Dataset. Our framework is adaptable to different real-world datasets. One can import real datasets from his application into our system by converting them into the standard format, which is a set of feature vectors for the data items and their relevance scores. From the provided feature vectors, the dissimilarity between the data items can be computed using existing distance measures (e.g. Euclid, Jaccard, Hamming, fuzzy, and categorical [3]). The dissimilarities are then fed along with the relevance scores to the diversification algorithms in the computing layer. In the following, we discuss two real-world datasets and the procedure to convert them to the standard format.

- *Camera dataset* [11]. The Camera dataset contains information about 497 cameras with 8 attributes per camera such as price, #megapixels and brand.These attributes are used to calculate the dissimilarity between the cameras (using Hamming distance) while the relevance of a camera in the dataset is computed from the price attribute. The referenced diversity for evaluating an algorithm output is measured by the number of brands it contains (see subtopic recall metric).
- *TREC dataset.* We also evaluate the algorithms on the ClueWeb12 dataset which is used in various TREC tracks [7]. More specifically, we leverage the freely-accessible portion of the dataset provided at [1]. It contains 17796 web pages for 50 queries. In order to use the TREC dataset in our framework, a preprocessing step is required to build the feature vectors and relevance scores for the web pages. First, we model the textual information of the web pages by tf-idf feature vectors. As the number of features is high, we also perform dimensionality reduction using latent semantic analysis to keep 50 most significant features. Then, we calculate the relevance scores of the web pages w.r.t a query using the baseline run of the TREC web track [7]. Finally, since the subtopics of the web pages are not available, we perform clustering based

on the feature vectors to group the web pages into 20 clusters (subtopics). It is worth noting that our evaluation results on other test collections (e.g. NCTIR [16], CLEF [4]) are similar and omitted for brevity sake.

Synthetic Data. Synthetic data is generated from the simulation module to help benchmark users to study unbiased evaluations of diversification techniques in a wide range of scenarios. To this extent, we vary the five parameters $\langle n, m, \sigma, \delta, \theta \rangle$: (i) n – the number of data items, (ii) m – the number of subtopics, reflecting the possible characteristics of original data, (iii) σ – the relevance difference between subtopics, reflecting the spectrum of relevance scores of data items, (iv) δ – the subtopic distance, reflecting the dissimilarity between items in different subtopics (the higher the distance between subtopics, the more dissimilar between their items), and (v) θ – the subtopic density difference, reflecting the spectrum of number of data items in the subtopics. The importance of these parameters to the diversification problem is described in Sect. 3.3.

Technically, we model data items as data points. The dissimilarity between items is measured by the Euclid distance between the corresponding points. They are partitioned into various clusters, where each cluster represents a subtopic that the associated items belong to. From this modeling, our simulation module generates synthetic data using the above parameters in the following steps:

1. First, we generate m cluster centroids such that the distance between two centroids is δ, which is varied in $[0, 1]$.
2. We calculate a set of density ratios α: $\{\frac{1}{m} - \lfloor \frac{m}{2} \rfloor \theta, \frac{1}{m} - (\lfloor \frac{m}{2} \rfloor + 1)\theta, ..., \frac{1}{m}, ..., \frac{1}{m} + (\lfloor \frac{m}{2} \rfloor - 1)\theta, \frac{1}{m} + \lfloor \frac{m}{2} \rfloor \theta\}$. The density ratios allow us to calculate the size of the clusters where the first density ratio $\alpha_1 = \frac{1}{m} - \lfloor \frac{m}{2} \rfloor \theta$ is associated with the first cluster and so on.
3. In the x-th cluster, generate $\approx n \times \alpha_x$ points gathering around the centroids (the total number of points is n and α_x is the density ratio of the x-th cluster)
4. Generate relevance scores such that items in the same cluster follow a normal distribution and the difference between the distribution means of two clusters is σ.
5. All relevance and dissimilarity values are then normalized into $[0, 1]$.

It is worth noting that the above simulation process can be customized as there are various parameters that can be changed in addition to the above parameters $\langle n, m, \sigma, \delta, \theta \rangle$. However, for the sake of simplicity, we limit ourselves to the above parameters. There are various design choices that we made regarding the simulation process. First, as each pair of data items has a dissimilarity value and the larger this value is, the more dissimilar between them, it is intuitive that we model the data items as data points and the distance between them as their dissimilarity values. Second, documents or data items that belong to the same subtopics are more similar to ones from different subtopics as they reflect the same aspects of the search keyword. As a result, data items that belong to a subtopic have smaller distance between them, hence, they form a cluster. This means each cluster represents a subtopic. Third, the number of documents

varies among subtopics as some subtopics are more popular than the others. This motivates us to vary the size of the subtopics using the subtopic density difference parameter θ. Lastly, among all the subtopics, at most one can express the user intention when searching for the keyword. As a result, some subtopics are more relevant than the others, which means the relevance values vary among the subtopics. We simulate this observation using the relevance difference parameter σ.

3.3 Evaluation Procedure

For comparative evaluation, we use two well-known metrics: *normalized relevance* [20] and *subtopic recall* [25].

- **Normalized relevance:** indicates how well an algorithm preserves relevance when diversity is taken into account. The normalized relevance of a subset $D^* \subset D$ of k items is defined as the sum of their relevance scores over the sum of k highest relevance scores:

$$nRev(D^*) = \frac{\sum_{d_i \in D^*} r_i}{\max_{D_k \subseteq D, |D^k|=k} \sum_{d_i \in D^k} r_i}$$

- **Subtopic recall:** reflects the actual degree of diversity of resulting items. Formally, the subtopic recall of an item set D^* is calculated as the proportion of the number of subtopics it covers over the total number of subtopics in the original item set D:

$$tRec(D^*) = \frac{num\_of\_subtopics(D^*)}{num\_of\_subtopics(D)}$$

For deep understanding, we characterize the diversification methods implemented in the benchmark using six different measures:

- **Computation time:** is an important measure, as every system has limited resources. It helps to choose the right techniques for particular applications under time constraints.
- **Effect of #displayed items:** in practice, it is common that users want to refine the result set by changing the number of displayed items they want to see. As the displayed budget increases, we expect that algorithms are able to cover more subtopics and include more relevant items into the result set. To validate this hypothesis, we need to examine the effect of number of displayed items to the diversification result.
- **Stability:** this property is important to support incremental data exploration. If a user is first presented with the top-10 data items, but then extends the result to the top-20, the expectation is clearly that the top-10 remain unchanged. In other words, the algorithms have to be stable in the sense that the result set can be extended in size to support a user in drilling down into data items. Therefore, there is a need to analyze the stability of the algorithms as the number of displayed items changes.

- **Effect of #subtopics:** the number of subtopics is a common measure to capture the degree of diversity. Since a dataset might cover a large number of subtopics, including all of them in a subset of output data items is challenging. To understand the behavior of diversification techniques, we need to study the effects of varying the number of subtopics in data.
- **Effects of relevance distribution:** beside diversity, relevance is another aspect that determines the course of the diversification problem. For example, if the relevance scores of data items are very similar to each other, relevance becomes a less significant aspect; and thus, algorithms that select data items regardless of their relevance scores might become the winners. To validate this intuition systematically, we examine wide-ranging configurations of the relevance difference parameter σ when generating the synthetic data.
- **Effects of dissimilarity distribution:** another factor that affects diversification is the distribution of dissimilarity values between data items. Algorithms that mainly focus on relevance might become the preferred ones if these dissimilarity values are small. Therefore, it is necessary to analyze the sensitivity of diversification algorithms to the dissimilarity distribution. To this end, we vary the distance parameter δ to obtain various distributions and then compare the algorithms case by case.
- **Effects of subtopic density difference:** the density of the subtopics is an important factor that has a significant effect on the diversification results. For instance, if the difference between subtopic density is high, an algorithm that does not focus on diversifying may include most data items from the densest subtopics while ignoring ones from sparser subtopics. As a result, there is a need to examine the effect of subtopic density difference θ on the performance of diversification techniques.

For the purposes of simplifying understanding and easing comparison, we intentionally restrict the study here to the highlighted metrics and measures. But recall that our benchmarking framework is extensible. In that, various metrics such as Expected Reciprocal Rank [6], α-nDCG [8], and harmonic mean can easily be added. Interested readers can find further details, potential extensions and updates on our website[3].

3.4 Benchmarking Tool

Our benchmarking tool is designed with three main user-interface panels, namely *configuration*, *quantitative evaluation*, and *qualitative evaluation*.

- *Configuration.* This panel offers benchmark users the flexibility to configure our framework to their own needs. Users can import their own datasets or customize simulation factors to generate synthetic data. Users can select the diversification algorithms and evaluation metrics they want to compare.

[3] https://code.google.com/p/diversity-benchmark.

- *Quantitative Evaluation.* The quantitative evaluation panel shows the numeric results plotted through the evaluation metrics. In that, user is able to observe the effects of configured parameters on the performance of diversification algorithms. More importantly, we offer a cross-metric comparison between diversification algorithms. By simultaneously displaying the evaluation results of different metrics, our system allows user to select a well-suited algorithm for his particular applications.
- *Qualitative Evaluation.* We also provide a data visualization in 3D view (when the synthetic data is two-dimensional), in which the XY-axes capture the distance (dissimilarity) between data items whereas the Z-axis represents the relevance score. Users can rotate, zoom in, zoom out the view to analyze different aspects of the results. Through the visualization, users can check the distribution of the output data items and qualitatively evaluate their coverage over the subtopics of original data.

Interested users can download the tool from our website[4] to test the above features.

4 Experimental Evaluation

Now we proceed to report benchmarking results, which ran on a CPU 2.8 GHz - 4GB RAM system. The main goal of the experiments is not only to compare the diversifying performances, but also to analyze the effects of configurable parameters on the performance behavior. Due to space limitations, further details can be found on our website[5].

4.1 Computation Time

We study computation time on different configurations of the number of input items n and the displayed budget number k. Table 2 shows the computation time of each technique, averaged over 100 runs, when varying $n \times k$ from 100×10 to 500×30.

In general, each algorithm category has its own winner on this concern (less than 100ms for all settings). This result is straightforward to understand – these techniques only need one iteration to select an item and do not execute expensive routines. In contrast, GrassHopper and MSD have the highest running time. For example, with $n = 500$ and $k = 20$, GrassHopper runs nearly 6s, MSD needs 272ms while the others require less than 100ms. This is because MSD finds two data items in each iteration while GrassHopper needs multiple iterations that involve complex matrix operations to compute the stationary distribution and absorption probabilities.

[4] https://code.google.com/p/diversity-benchmark.
[5] https://code.google.com/p/diversity-benchmark.

Table 2. Average computation time (ms) over 100 runs (the lower, the better)

$n \times k$ *	Motley	Swap	MSD	MMR	AG	GrassHopper
100×10	2	4	8	3	1	29
100×20	2	7	11	4	1	50
100×30	2	12	13	8	1	66
200×10	7	14	25	8	6	213
200×20	7	21	41	13	7	393
200×30	7	29	56	22	7	564
500×10	42	56	160	48	43	3029
500×20	42	68	272	63	45	6062
500×30	42	84	380	86	46	8943

* $n \times k$: n data items and k results

4.2 Effect of Number of Displayed Items

Figure 2 illustrates the effects of varying the displayed budget number from $k = 2$ to $k = 20$ on the performance of the diversification algorithms. The experiment is conducted on both real-world and synthetic data and due to space constraint, we only report the results on the real-world datasets. Regarding the TREC dataset, since it contains 50 different queries, we calculate the subtopic recall and normalized relevance values for each query and report the result as the average over 50 queries.

A key finding is that as the number of displayed items increases, the subtopic recall values also increase. For example, when k increases from 2 to 20, the subtopic recall of GrassHopper increases from 0.1 to 0.4 for the TREC dataset

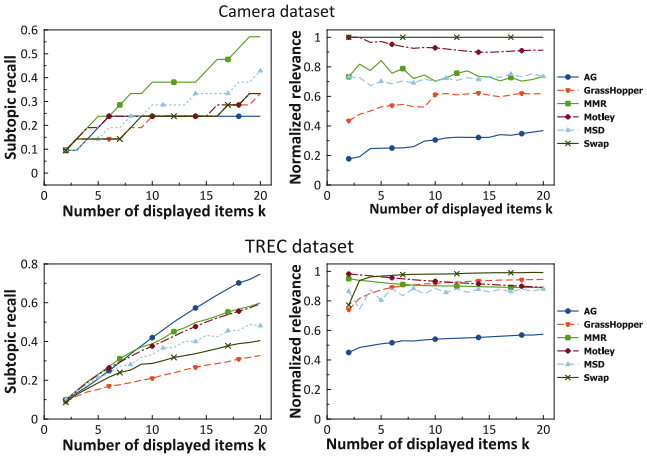

Fig. 2. Effects of number of displayed items k

and from 0.1 to 0.3 for the Camera dataset. This is expected and can be explained as follows. Since the number of displayed items increases, the algorithms are able to include more items into the result set. These items may belong to existing or new subtopics, which makes the subtopic recall values increase. Another interesting observation is the performance of MSD w.r.t both datasets as its subtopic recall and normalized relevance follow a zigzag pattern. The reason is that MSD picks two data items as a time. When the number of displayed items is odd, MSD needs to select the last data item randomly, which incurs in the decrease of both subtopic recall and normalized relevance.

4.3 Stability of the Algorithms

In order to check the stability of the algorithms, for both real and synthetic dataset, we verify whether the output set $D^*_{k_1}$ is a subset of $D^*_{k_2}$ if $k_1 < k_2$ as we increase the number of displayed items k. If this proposition is not satisfied for any dataset or any value of k, we conclude that the algorithm is not stable.

An important finding is that only Swap, Motley and MSD are not stable. For Swap algorithm, since it initializes the result set by the top-k relevant data items, the initial result set is different for different values of k. This affects subsequent swaps as the items to be swapped are selected based on the current result set. Since the swap sequences are different w.r.t k, Swap algorithm is not stable. Regarding Motley and MSD, as they may add items to the result set randomly, there are cases that $D^*_{k_1} \not\subset D^*_{k_2}$. Other algorithms such as MMR, AG and GrassHopper, as they build the result set by adding one element at a time and do not involve any randomness or depend on current result set, are stable.

4.4 Effects of #subtopics

Figure 3 depicts the result of this experiment, which was conducted by varying the number of subtopics m from 2 to 8. The number of input items n increases from 200 to 800 as the number of subtopics increases. The displayed budget number k is fixed to 10, which is often the default number of many well-known applications (e.g. google.com, bing.com). As this experiment requires changing the number of subtopics, which can only be achieved on synthetic data, this experiment is not conducted on the real datasets.

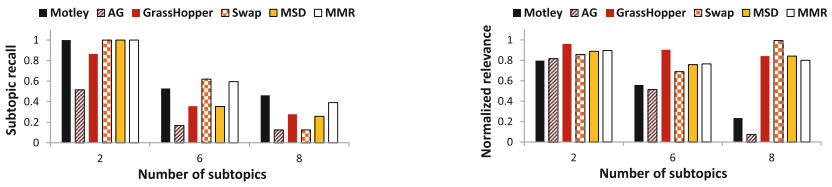

Fig. 3. Effects of the number of subtopics

In general, we observe a reduction in subtopic recall as m increases. For example, the subtopic recall of GrassHopper decreases from 0.8 to 0.2 while that of Swap decreases from 1 to 0.1 as m increases from 2 to 8. The reason is that the number of subtopics covered by the algorithms does not increase linearly with the number of subtopics in the dataset. Another important remark is that MMR can balance between diversity and relevance despite the changes in the number of subtopics. More specifically, MMR ranks second regarding subtopic recall and third with respect to normalized relevance. This can be explained as follows: in the beginning, when the result set D^* contains few items, the relevance term in the objective function of MMR is the dominant term, which makes MMR to select items with high relevant scores. In contrast, when D^* has many items, the diversity term becomes dominant, which forces MMR to find diverse items.

4.5 Effects of Dissimilarity Distribution

The empirical analysis is illustrated in Fig. 4, in which the subtopic distance δ varies from 0.05 to 0.25. We fix the number of subtopics $m = 5$, the displayed budget number $k = 10$, and the number of input items $n = 500$. This experiment is only conducted on synthetic data as it requires changing the subtopic distance δ.

An interesting observation is that the normalized relevance values of all algorithms tend to decrease while their subtopic recall values remain the same as the subtopic distance δ increases. For instance, the normalized relevance value of AG decreases from 0.4 to 0.1 while its subtopic recall stays at 0.2. This can be explained as follows: since the dissimilarity between items increases as we increase δ, the algorithms are able to select items from low relevant subtopics as the loss in relevance is offset by the gain in dissimilarity. Another interesting observation is the performance of Motley. While its subtopic recall values are the highest (over 0.9) and increase with the subtopic distance, its normalized relevance values are the second lowest among the algorithms. The reason lies in its computing model. Motley are able to cover most subtopics since it explicitly aims to maximize diversity by selecting an item only if it is dissimilar with other items in the result set. In addition, since it traverses the candidate items in descending order of relevance, many highly relevant items are not selected if they belong to the same subtopic with the highest relevant item, thus low normalized relevance values.

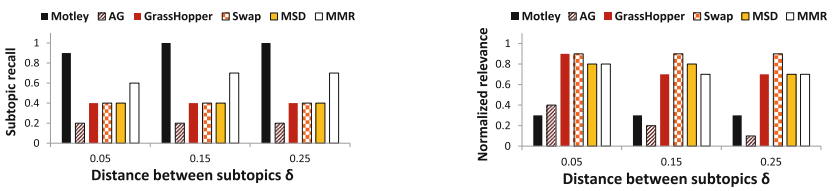

Fig. 4. Effects of dissimilarity distribution

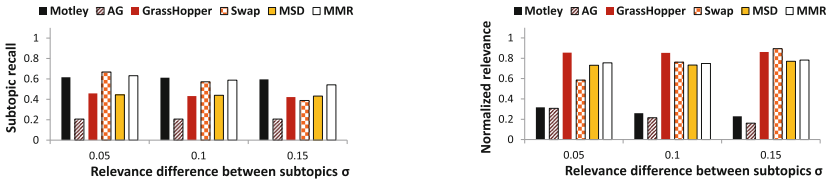

Fig. 5. Effects of relevance variance between domain topics

4.6 Effects of Relevance Distribution

Figure 5 illustrates the effects of varying the relevance difference between the subtopics from $\sigma = 0.05$ to $\sigma = 0.15$ and fixing the number of subtopics $m = 5$, the displayed budget number $k = 10$, and the number of input items $n = 500$. Since the relevance difference between the subtopics can not be changed in the real-world dataset, we only perform this experiment on the synthetic dataset.

A noticeable observation is the tradeoff between diversity and relevance of the algorithms. This can be seen clearly through the performance of Swap. It has high subtopic recall and low normalized relevance when σ is small but the situation is reversed when σ is high. The reason is that Swap is highly sensitive to the relevance drop threshold that a swap can occur. When σ is lower than this threshold (i.e. $\sigma < 0.1$), Swap is able to diversify the result set by exchanging high relevant items in the current result set with low relevant ones as these swaps do not violate the constraint. However, when the relevance difference σ is high, Swap tries to retain highly relevant items since swapping these items out would violate the constraint. Another interesting observation is that the subtopic recall of MSD remains the same at 0.4 as σ increases. Since MSD picks two items at a time, these items tend to be from a pair of subtopics with the highest dissimilarity, which explains the constant subtopic recall.

4.7 Effects of Subtopic Density Difference

The experimental results are shown in Fig. 6, in which the subtopic density difference θ varies from 0.05 to 0.08. We fix the number of subtopics $m = 5$, the displayed budget number $k = 10$, and the number of input items $n = 500$.

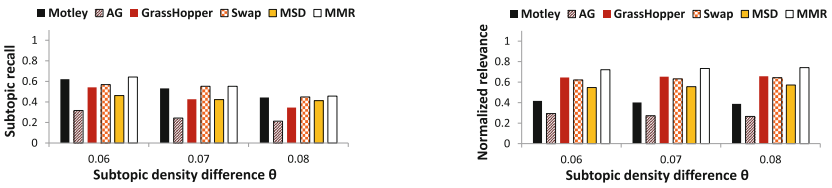

Fig. 6. Effects of subtopic density difference

Since this experiment requires changing the subtopic density difference θ, it is conducted on the synthetic data.

In general, we observe that as θ increases, the subtopic recall decreases while normalized relevance values change marginally. For example, the subtopic recall of Motley decreases from 0.62 to 0.44 while its relevance values remain around 0.4 as we vary θ. The reason behind this observation is that with higher values of θ, some subtopics contain a larger number of data items in comparison with other subtopics. As a result, the data items in smaller subtopics may not be included to the final result. On the other hand, the algorithms are still able to select highly relevant data items from large subtopics which makes the relevance values change very slightly.

5 Conclusions and Future Work

This paper presented a thorough evaluation and comparison of diversification techniques widely used in various domains. We offered an overview of three major classes (threshold-based, function-based and graph-based) of diversification techniques, while discussing about the characteristics of their underlying models. We then introduced the component based benchmarking framework, in which a new diversification technique as well as a new measurement or a dataset can be easily plugged. During the framework development, we made the best effort to re-implement and integrate the most representative diversification techniques, and evaluated them in a fair manner. We also analyzed various performance factors for each technique, including computation time, effects of number of subtopics, number of displayed items, dissimilarity distribution, relevance distribution and stability of the algorithms.

We here summarize the principal findings as a set of guidelines for how to choose appropriate diversification techniques on the following scenarios:

- Overall, MMR performs best in terms of multiple criteria. It has low computation time while it can find relevant items belonging to different subtopics.
- For applications that require fast computation, Motley and AG are the winners. But the two other runner-ups, Swap and MMR, are not significantly slower.
- For applications that focus on diversity, Motley should be used as it can return the highest number of subtopics.
- For datasets whose dissimilarity between subtopics is high, Motley is the best choice in terms of diversity while MMR is the best in terms of balance between diversity and relevance.
- For datasets that have high variance of relevance distribution, we suggest using MMR. In contrast, Swap is the best if the relevance difference is low and diversity is the main concern.
- For datasets in which the number of data items between subtopics vary widely, MMR is the best choice in terms of diversity and relevance.

– For applications that the number of displayed items can be changed, we suggest not to use Swap, Motley and MSD as these algorithms are not stable. Among stable algorithms, MMR and AG are both good choices in terms of diversity.

Category	Winner	2nd best	Worst
Computation time	**Motley**	AG	GrassHopper
Number of subtopics	**MMR**	Swap	AG
Dissimilarity distribution	**MMR**	Motley	AG
Relevance distribution	**MMR**	Swap	AG
Subtopic density difference	**MMR**	Swap	AG
Number of results[a]	**MMR**	AG	GrassHopper

[a]Only stable algorithms are considered.

As a concluding remark, we recommend potential applications to use our benchmarking framework as a tool to find out the well-suited diversification techniques accordingly. As the benchmark source code as well as the datasets used in the benchmark are publicly available, we expect that the experimental results presented in this paper will be refined and improved by the research community, in particular when more data become available, more experiments are performed, and more techniques are integrated into the framework.

Acknowledgment. The research has received funding from the ScienceWise project.

References

1. http://boston.lti.cs.cmu.edu/clueweb12/TRECcrowdsourcing2013/
2. Agrawal, R., et al.: Diversifying search results. In: WSDM, pp. 5–14 (2009)
3. Boriah, S., et al.: Similarity measures for categorical data: a comparative evaluation. In: SIAM, pp. 243–254 (2008)
4. Braschler, M.: CLEF 2001 - overview of results. In: Peters, C., Braschler, M., Gonzalo, J., Kluck, M. (eds.) CLEF 2001. LNCS, vol. 2406, p. 9. Springer, Heidelberg (2002)
5. Carbonell, J., et al.: The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: SIGIR, pp. 335–336 (1998)
6. Chandar, P., et al.: Preference based evaluation measures for novelty and diversity. In: SIGIR, pp. 413–422 (2013)
7. Clarke, C.L., et al.: Overview of the trec 2009 web track. Technical report, DTIC Document (2009)
8. Clarke, C.L., et al.: Novelty and diversity in information retrieval evaluation. In: SIGIR, pp. 659–666 (2008)
9. Deng, T., et al.: On the complexity of query result diversification. Proc. VLDB Endowment **6**, 577–588 (2013)
10. Drosou, M., et al.: Search result diversification. ACM SIGMOD Rec. **39**, 41–47 (2010)

11. Drosou, M., et al.: Disc diversity: result diversification based on dissimilarity and coverage. Proc. VLDB Endowment **6**, 13–24 (2012)
12. Gabrilovich, E., et al.: Newsjunkie: providing personalized newsfeeds via analysis of information novelty. In: WWW, pp. 482–490 (2004)
13. Gollapudi, S., et al.: An axiomatic approach for result diversification. In: WWW, pp. 381–390 (2009)
14. Hasan, M., et al.: User effort minimization through adaptive diversification. In: KDD, pp. 203–212 (2014)
15. Jain, A., Sarda, P., Haritsa, J.R.: Providing diversity in k-nearest neighbor query results. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 404–413. Springer, Heidelberg (2004)
16. Kando, N., et al.: Overview of IR tasks at the first NTCIR workshop. In: NTCIR, pp. 11–44 (1999)
17. Küçüktunç, O., et al.: Diversified recommendation on graphs: pitfalls, measures, and algorithms. In: WWW, pp. 715–726 (2013)
18. Rafiei, D., et al.: Diversifying web search results, pp. 781–790. In: WWW 2010 (2010)
19. Skoutas, D., et al.: Tag clouds revisited. In: CIKM, pp. 221–230 (2011)
20. Tong, H., et al.: Diversified ranking on large graphs: an optimization viewpoint. In: KDD, pp. 1028–1036 (2011)
21. Vaughan, L., et al.: Search engine coverage bias: evidence and possible causes. Inf. Process. Manag. **40**, 693–707 (2004)
22. Verheij, A., et al.: A comparison study for novelty control mechanisms applied to web news stories. In: WI, pp. 431–436 (2012)
23. Vieira, M.R., et al.: On query result diversification. In: ICDE, pp. 1163–1174 (2011)
24. Yu, C., et al.: It takes variety to make a world: diversification in recommender systems. In: EDBT, pp. 368–378 (2009)
25. Zhai, C.X., et al.: Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In: SIGIR, pp. 10–17 (2003)
26. Zhang, B., Li, H., Liu, Y., Ji, L., Xi, W., Fan, W., Chen, Z., Ma, W.Y.: Improving web search results using affinity graph. In: SIGIR, pp. 504–511 (2005)
27. Zhu, X., et al.: Improving diversity in ranking using absorbing random walks. In: NAACL, pp. 97–104 (2007)
28. Ziegler, C.N., et al.: Improving recommendation lists through topic diversification. In: WWW, pp. 22–32 (2005)