Selecting Representative and Diverse Spatio-Textual Posts over Sliding Windows

Dimitris Sacharidis
TU Wien
dimitris@ec.tuwien.ac.at

Paras Mehta
Freie Universität Berlin
paras.mehta@fu-berlin.de

Dimitrios Skoutas
Athena R.C.
dskoutas@imis.athena-innovation.gr

Kostas Patroumpas
Athena R.C.
kpatro@imis.athena-innovation.gr

Agnès Voisard
Freie Universität Berlin
agnes.voisard@fu-berlin.de

ABSTRACT

Thousands of posts are generated constantly by millions of users in social media, with an increasing portion of this content being geotagged. Keeping track of the whole stream of this spatio-textual content can easily become overwhelming for the user. In this paper, we address the problem of selecting a small, representative and diversified subset of posts, which is continuously updated over a sliding window. Each such subset can be considered as a concise summary of the stream's contents within the respective time interval, being dynamically updated every time the window slides to reflect newly arrived and expired posts. We define the criteria for selecting the contents of each summary, and we present several alternative strategies for summary construction and maintenance that provide different trade-offs between information quality and performance. Furthermore, we optimize the performance of our methods by partitioning the newly arriving posts spatio-textually and computing bounds for the coverage and diversity of the posts in each partition. The proposed methods are evaluated experimentally using real-world datasets containing geotagged tweets and photos.

CCS CONCEPTS

• **Information systems → Spatial-temporal systems**; **Data streaming**; **Information retrieval diversity**;

KEYWORDS

spatio-textual streams, continuous diversification, summarization

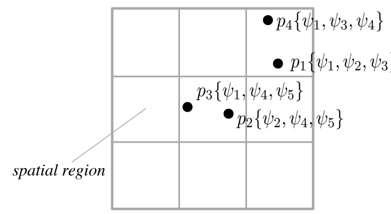


Figure 1: Running example of spatio-textual posts.

1 INTRODUCTION

Millions of users generate content daily in social media and other online platforms. A large portion of this content is geotagged, and can thus be modeled as a *stream* of spatio-textual posts, comprising a *geolocation* and a set of *keywords* (e.g., tags, hashtags). Typical examples include geotagged tweets and photos, news articles with location references, user check-ins at certain places, reviews and comments about various Points of Interest (POI), etc. This combination of textual content with geospatial information can be a valuable source for a variety of tasks, such as identifying and monitoring events at various locations [14], recommendation systems based on spatial-keyword subscriptions [31], mining trending topics in different areas [24], studying the spatial distribution of opinions and sentiments associated with various POIs, and more.

Nevertheless, keeping track of the entire stream is *impractical*, since it can easily become overwhelming for the user. It is also *undesirable*, as such content is typically characterized by a high degree of redundancy, e.g., due to identical news articles or similar stories being published by several outlets [32], same posts re-tweeted by multiple users, or similar opinions and comments being expressed. Instead, it is easier and often sufficient to draw conclusions and to identify interesting trends, behaviors and patterns by only examining a much smaller, *representative* subset of the stream. Since new content is continuously being produced, while past items lose their significance with time, such summaries need to be continuously maintained and updated as the underlying content evolves. Being able to do so efficiently and incrementally in a scalable manner can benefit search and recommendation engines as well as publish/subscribe systems, since users can obtain a concise and summarized overview of the underlying information, including appropriately selected anchor posts for further exploration.

In this paper, we consider four criteria for effectively summarizing a set of spatio-textual posts. Specifically, a summary should have (a) high *textual coverage*, i.e., capture the most representative textual information; (b) high *spatial coverage*, i.e., capture the most representative spatial information; (c) high *textual diversity*, i.e., include posts textually dissimilar to each other; and (d) high *spatial diversity*, i.e., include posts with locations far from each other.

Example 1.1. Figure 1 depicts a running example used throughout the paper to illustrate the described concepts and our approach. Consider four posts p_1, p_2, p_3, p_4, drawn as point locations on the

SSDBM '18, July 9–11, 2018, Bozen-Bolzano, Italy

ACM ISBN 978-1-4503-6505-5/18/07…$15.00
https://doi.org/10.1145/3221269.3221290

2D space, which is partitioned using a uniform grid. Every post is associated with a set of keywords. We assume that each post contains three keywords, drawn from a vocabulary $\{\psi_1, \psi_2, \psi_3, \psi_4, \psi_5\}$. Now, assume that the goal is to summarize these posts by selecting only two of them. In terms of textual coverage, summary $S_1 = \{p_1, p_3\}$ is better than $S_2 = \{p_2, p_3\}$, because S_1 covers all five keywords, whereas S_2 covers only four. Similarly, with respect to spatial coverage, summary S_1 covers two cells in the grid compared to one cell covered by S_2. Regarding textual diversity, summary S_1 is more dissimilar than S_2, as the two posts in S_1 share only one keyword, as opposed to two shared by posts in S_2. Finally, summary S_1 also exhibits higher spatial diversity than S_2, as the spatial distance between posts in S_1 is larger than that of posts in S_2. □

On the one hand, certain parts of the textual and spatial information may be deemed more relevant or significant than others depending on application-specific criteria (e.g., frequency, popularity). This can be reflected in the summary by favoring selection of posts that contain such attributes. In our approach, this is achieved via the criterion of *coverage*. Intuitively, the aim of textual and spatial coverage is to ensure that the summary reflects as much as possible the current textual and spatial content of the stream.

On the other hand, Web and user-generated content is inherently repetitive and redundant, which makes a coverage-based summary prone to biases, and thus less useful for exploratory search and browsing. Diversifying search results is an established and well-studied concept in information retrieval and web search, and can significantly improve the quality of produced results and thus increase user satisfaction in several practical applications [8, 12, 30]. In our approach, this is achieved via the criterion of *diversity*. Intuitively, the aim of textual and spatial diversity is to increase the variety within the summary, favoring a mixture of differing topics, aspects, opinions or sentiments, and thus reducing bias.

Note that the criteria of coverage and diversity can be contradictory. Yet, it is possible to account for both of them and allow for different tradeoffs, using various diversification models and formalisms proposed in the literature. Finding the optimal diversified subset of posts under these formulations is NP-hard problem. In practice, heuristics are employed to compute approximate solutions more efficiently [12]. Nonetheless, existing approaches are either inapplicable to or inefficient for our problem setting.

First, almost all existing approaches only address *static* settings (i.e., computing a diversified subset of a given document collection or of the result set of a given query), whereas very few ones have considered a dynamic or *streaming* context [9, 23]. Moreover, those that work over streams typically employ various restrictions and assumptions that simplify the problem, making it easier to tackle, but at the expense of restricting flexibility and generality.

Second, regardless of static or streaming context, the particular type of contents in the handled objects (posts in our case) is considered an orthogonal issue. Thus, proposed solutions concern only the generic diversification algorithm itself, without further optimizing the process with specific coverage/relevance scores and dissimilarity measures taking into account the type of posts.

In this paper, we attempt to fill these gaps by computing and continuously updating textually and spatially diversified summaries over streams of *spatio-textual* posts. Our approach adopts the *sliding window* model, and is based on examining successive chunks of the incoming stream to incrementally update the summary as new posts arrive and old ones expire. First, we present several diversification strategies that provide different trade-offs between maximizing the quality (i.e., combined coverage and diversity) of the resulting subset and minimizing computational cost. Then, we focus on the specific case of spatio-textual posts, proposing optimizations that can be applied to enhance the efficiency of those diversification strategies. To the best of our knowledge, our work is the first to address the problem of maintaining a diversified summary of posts over a stream of spatio-textual posts.

Our approach is well-suited for exploratory search. The user is not required to select a specific location or to specify a set of relevant keywords. Instead, the goal is to provide an overview of the whole content in such a way that both reflects most popular and frequent items while still providing a sufficient degree of variety to facilitate further exploration. This is particularly useful when browsing social media content, where the user seeks to find interesting events, trends and topics, but without knowing a priori what this information is about or where it occurs or refers to. Note that a preliminary investigation of this problem has been presented in [26]. In this paper, we further elaborate on this problem, providing an in depth investigation, suggesting and evaluating more advanced summarization techniques.

Our main contributions can be summarized as follows:

- We formally define the problem of summarizing a stream of spatio-textual posts over a sliding window, defining specific spatio-textual criteria of coverage and diversity.
- We show how our formulation is related to the max-sum diversification problem, and we remark that there exists a simple greedy algorithm that can provide a 2-approximate solution to the general static problem in linear time, instead of a well-known quadratic time algorithm.
- We propose different streaming summarization strategies that can trade result quality against computational cost.
- We optimize our proposed strategies with light-weight spatio-textual structures, which are used to update the result set efficiently upon each window slide.
- We present an experimental evaluation to study and compare the performance trade offs of our proposed methods and the resulting performance gains of our algorithms.

The rest of the paper is organized as follows. Section 2 reviews related work on summarization and diversification methods. Section 3 formally defines the problem and the criteria for spatio-textual coverage and diversity. Section 4 relates our problem to max-sum diversification and introduces a simple algorithm for the static problem. Section 5 presents our solution to the continuous summarization problem, while Section 6 discusses optimizations based on spatio-textual partitioning. Section 7 presents an experimental evaluation of our methods, and Section 8 concludes the paper.

2 RELATED WORK

We first present an overview of document summarization focusing on diversification-based techniques on static data, and then discuss the case of diversification over streaming data.

2.1 Summarization via Diversification

The seminal work in [3] studied the problem of document summarization and formulated it as an instance of the diversification problem, which was later studied in various incarnations. Diversification generally aims at reducing repetition and redundancy in the result set presented to the user by selecting the final set of results based not only on each document's relevance to the query, but also on the dissimilarity amongst the selected results. This increases the variety and novelty of information included in the diversified result set, which is particularly important for queries inherently ambiguous or targeting different subtopics, perspectives, opinions and sentiments. In such cases, diversification allows to better cover and represent these different aspects in the result set.

Different formulations for search results diversification exist (see [8, 12, 30] for an overview). According to the framework proposed in [12], given a query Q and an initial result set R containing all documents relevant to Q, the goal is to select a relatively small subset R^* of R, with $|R^*| = k$. Set R^* should maximize an objective function ϕ that combines: (a) a *relevance score* assessing how relevant each document in R^* is to the given query, and (b) a *diversity score* measuring how diverse the documents in R^* are to each other. Function ϕ can take different forms, such as *max-sum*, *max-min*, and *mono-objective* diversification. For instance, in the *max-sum* variant, ϕ is defined as the weighted sum of (a) the total relevance of the documents in R^* to Q, and (b) the sum of pairwise distances among the documents in R^*.

Typically, finding the exact result set that maximizes the diversification objective is NP-hard. Thus, approximate solutions usually rely either on *greedy* heuristics, which build the diversified set incrementally, or on *interchange* heuristics, which gradually improve upon a randomly selected initial set by swapping its elements with other ones that improve its diversity. In a recent work [4], the notion of approximate, composable core-sets has been used to address the k-diversity maximization problem in general metric spaces for Streaming and MapReduce processing models. A core-set is a small set of items that approximates some property of a larger set [17].

We note that other types of diversification problems have also been studied, such as taxonomy/classification-based diversification [1, 29] or multi-criteria diversification [7].

Moreover, summarization has also been studied in the context of coverage alone, without a notion of diversity [20, 28]. For the coverage problem in [10], the goal is to select the minimum subset of documents that have distance to each other at least ϵ, and cover the whole dataset, i.e., each non-selected document lies within distance at most ϵ from a selected one. However, the size of the summary is not fixed, but depends on the distance threshold ϵ.

Finally, the work in [32] aims at selecting k relevant and diverse posts in a spatio-temporal range. In this case, the spatial relevance of a post is defined by its proximity to the user's location. Instead, we define spatial importance in terms of spatial coverage, i.e., how many other posts exist in a post's neighborhood. In our view, this formulation is better suited for summarization, since a post may be spatially important even if it is far from the center of the studied spatial range (e.g., the user's location) as long as there are many other posts around it. More importantly, diversity in [32] is a hard constraint: returned posts must contain at least h different topics, so the result set could be empty. In contrast, we consider diversity as an extra factor in the objective function, which increases flexibility but also makes the problem computationally harder.

2.2 Diversification over Streaming Data

Few works so far have considered the problem of continuously maintaining a diversified result set over streaming data. The approach in [9] adopts the *max-min* objective function for diversification, which entails maximizing the minimum distance between any pair of documents in the result set. The proposed method relies on the use of *cover trees*, and provides solutions with varying accuracy and complexity for selecting items that are both relevant and diverse. Moreover, it introduces a sliding window model for coping with the continuous variant of the problem against streaming items. However, a cover tree needs to be incrementally updated by keeping all raw items within the current window, which can have a prohibitive cost in space and time when dealing with massive, frequently updated streaming data. In our case, we rely on much more lightweight aggregate information to speed up the computation of the new result set every time the window slides.

The work presented in [23] assumes a *landmark window* model, i.e., a window over the stream that spans from a fixed point in the past until the present. The proposed online algorithm checks every fresh document against those in the current result set, and performs a substitution if it increases the objective score of the set. Thus, this method addresses the diversification problem on an ever increasing stream of objects. However, it is restricted by the fact that it always considers exactly one incoming document, and does not handle document expirations. In Section 5.2.2, we revisit this algorithm and introduce its adaptation involving sliding windows, i.e., where both the start and the end points of the window slide.

Recently, there has been an increasing interest in publish/subscribe systems. In this setting, users subscribe to queries, and the system has to deliver incoming messages to all users for which the incoming message matches their query. The main goal is to group similar subscriptions together, in order to minimize the computations required to identify all users relevant to each incoming message. Diversifying delivered messages to users applies also to this setting. Nevertheless, to the best of our knowledge, the criterion of diversity has so far been considered only in [5]. This work addresses the problem of diversity-aware top-k subscription queries over textual streams. Qualifying documents are ranked by text relevance, temporal recency, and result diversity according to respective score functions. Although very efficient, the proposed method is rather restrictive, as each new document is compared only to the oldest one in the current result set. If it improves the objective score, the replacement is made, otherwise the document is discarded. Moreover, the considered documents do not have a spatial attribute. In contrast, there exist some recent works on spatial-keyword publish/subscribe systems, which however do not consider diversity as a criterion [6, 16, 18, 31].

3 PROBLEM DEFINITION

Spatio-Textual Stream. We consider geotagged messages posted by users of social networks or microblogs (e.g., Facebook, Twitter, Flickr, Foursquare) in a streaming fashion, as defined next.

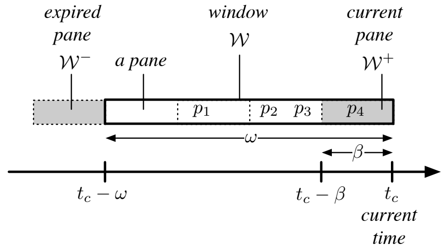


Figure 2: Time-based sliding window over a stream of posts.

Definition 1. *A spatio-textual post $p = \langle \Psi, \ell, t \rangle$ contains a set of keywords Ψ from a vocabulary V, and was generated at location ℓ (with coordinates (x, y)) at timestamp t.* □

We assume that posts become available as a stream of tuples, and we adopt a time-based *sliding window* model [19] over the stream.

Definition 2. *A time-based sliding window $\mathcal{W}$ is specified with two parameters: (a) a* range *spanning over the most recent ω timestamps backwards from current time t_c, and (b) a* slide step *of β timestamps. Upon each slide, window $\mathcal{W}$ moves forward and provides all posts generated during time interval $(t_c - \omega, t_c]$. These posts comprise the current* state *of the window, i.e.,*

$$\mathcal{W} = \{p : p.t \in (t_c - \omega, t_c]\} \tag{1}$$

Posts with timestamps earlier than $t_c - \omega$ are called expired. □

Such a window is illustrated in Figure 2. Once this window slides forward to current time, fresh posts are admitted into the state, while expired posts get discarded.

Spatio-Textual Summary. Our goal is to select an appropriate subset S of posts in window $\mathcal{W}$ to form an informative summary. In accordance with work on document summarization [11, 20], given a constraint on the maximum summary size, our objective is to construct a summary that *covers* as much as possible the entire set of posts in the current window while at the same time containing *diverse* information as much as possible. Formally, we capture these two requirements using the two measures defined next.

Definition 3. *The* coverage *$cov(S)$ of a summary S captures the degree to which the posts in the summary approximate the spatial and textual information in the window. We use a weight α to balance the relative importance of the two information facets:*

$$cov(S) = \alpha \cdot cov^T(S) + (1 - \alpha) \cdot cov^S(S). \tag{2}$$

□

Similar to [20], we define the *textual coverage* of a summary as

$$cov^T(S) = \sum_{p_i \in \mathcal{W}} \sum_{p_j \in S} sim^T(p_i, p_j) \tag{3}$$

where $sim^T(\cdot, \cdot)$ is a textual similarity metric between posts.

For our purposes, we consider the vector space model, and define $sim(\cdot, \cdot)$ as the *cosine similarity* of the vector representations of the posts. Specifically, each space coordinate corresponds to a keyword, and the vector's coordinate contains a weight representing the importance of the corresponding keyword relative to the window. While any *tf-idf* weighing scheme [22, 27] is possible, here we simply use term frequency and normalize the vectors to unit norm.

	p_1	p_2	p_3	p_4
p_1	1	1/3	1/3	2/3
p_2	1/3	1	2/3	1/3
p_3	1/3	2/3	1	1/3
p_4	2/3	1/3	2/3	1

(a) Textual Content

	p_1	p_2	p_3	p_4
p_1	1	0	0	1
p_2	0	1	1	0
p_3	0	1	1	0
p_4	1	0	0	1

(b) Spatial Content

Figure 3: Content similarities of posts in Figure 1.

	p_1	p_2	p_3	p_4
p_1	0	2/3	2/3	1/3
p_2	2/3	0	1/3	2/3
p_3	2/3	1/3	0	2/3
p_4	1/3	2/3	1/3	0

(a) Textual Content

	p_1	p_2	p_3	p_4
p_1	0	0.15	0.20	0.05
p_2	0.15	0	0.05	0.15
p_3	0.20	0.05	0	0.25
p_4	0.05	0.15	0.25	0

(b) Spatial Content

Figure 4: Content distances of posts in Figure 1.

Therefore, the textual coverage is computed as the sum over each pair of posts (one from the window and the other from the summary) of the inner product of their vector representations:

$$cov^T(S) = \sum_{p_i \in \mathcal{W}} \sum_{p_j \in S} \sum_{\psi} p_i^T[\psi] \cdot p_j^T[\psi], \tag{4}$$

where keyword ψ is used to index the vector and thus $p^T[\psi]$ denotes the normalized weight of keyword ψ of post p.

Example 3.1. Consider the vector space model for the five keywords in the example of Figure 1. Each post can be represented by a 5D normalized vector. For example, $p_1^T = (1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}, 0, 0)$, and $p_3^T = (1/\sqrt{3}, 0, 0, 1/\sqrt{3}, 1/\sqrt{3})$. Then, the textual similarity of two posts is given by the cosine similarity of their vectors, e.g., $sim^T(p_1, p_3) = 1/\sqrt{3} \cdot 1/\sqrt{3} = 1/3$. Figure 3a lists the cosine similarity for each pair of posts. Consider now the summary $S = \{p_1, p_3\}$ of the window $\mathcal{W} = \{p_1, p_2, p_3, p_4\}$. Its coverage can be computed as the sum of entries in the first and third rows of Figure 3a: $cov^T(S) = \sum_{p_i \in \mathcal{W}} sim^T(p_1, p_i) + \sum_{p_i \in \mathcal{W}} sim^T(p_3, p_i) = 14/3$. □

For the *spatial coverage*, we follow a similar formulation and define it as cosine similarity in a (different) vector space. Instead of keywords from a vocabulary, we have a set of *regions* from a predefined spatial partitioning ρ (e.g., regions could represent cells in a uniform grid). Intuitively, such a coarse partitioning allows for a macroscopic view of the posts in the window, where exact post locations are not important and thus coalesced into broader regions. As each post is always associated with a single region, the spatial content of a post is simply represented as a vector having a single weight 1 at the vector coordinate signifying the region containing the post's geotag. Thus, the spatial coverage is computed as:

$$cov^S(S) = \sum_{p_i \in \mathcal{W}} \sum_{p_j \in S} \left|\rho(p_i.\ell) = \rho(p_j.\ell)\right|, \tag{5}$$

where $\rho(\ell)$ is the region associated with location ℓ, and $|\rho(p_i.\ell) = \rho(p_j.\ell)|$ returns 1 if locations $p_i.\ell, p_j.\ell$ reside in the same region.

Example 3.2. Consider the vector space model for the nine spatial regions in the example of Figure 1. Each post is then represented as a 9-dimensional one-hot vector. For example, $p_1^S = (0, 0, 1, 0, 0, 0, 0, 0, 0)$, and $p_3^S = (0, 0, 0, 0, 1, 0, 0, 0, 0)$. Then the spatial similarity of two posts is 1 if the two posts reside in the same region, and 0 otherwise, e.g., $sim^S(p_1, p_3) = 0$. Figure 3b shows

S	cov^T	cov^S	div^T	div^T	f
p_1, p_2	14/3	4	2/3	0.15	2.37
p_1, p_3	14/3	4	2/3	0.2	2.38
p_1, p_4	15/3	2	1/3	0.05	1.85
p_2, p_3	14/3	2	1/3	0.05	1.76
p_2, p_4	15/3	4	2/3	0.15	2.45
p_3, p_4	15/3	4	1/3	0.25	2.40

Figure 5: Individual and objective scores for all 2-post summaries of the example in Figure 1.

the spatial similarity for each pair of posts. Similar to the case of textual coverage, the spatial coverage of summary $S = \{p_1, p_3\}$ can be computed as the sum of the entries in the first and third rows of Figure 3b: $cov^S(S) = 4$. □

Definition 4. *The* diversity $div(S)$ *of a summary S captures the degree to which the posts in S carry dissimilar information. As before, diversity is defined as the weighted sum of a textual and spatial term:*

$$div(S) = \alpha \cdot div^T(S) + (1 - \alpha) \cdot div^S(S) \tag{6}$$

□

Textual diversity is defined with respect to the vector space model. Specifically, textual diversity is the sum of cosine distance between all pairs of posts in the summary:

$$div^T(S) = \sum_{\{p,p'\}:p \neq p' \in S} \left(1 - \sum_{\psi} p_i^T[\psi] \cdot p_j^T[\psi]\right) \tag{7}$$

On the other hand, *spatial diversity* is defined based on a spatial distance (e.g., Euclidean, haversine) between summary posts' exact locations:

$$div^S(S) = \sum_{\{p,p'\}:p \neq p' \in S} dist(p.\ell, p'.\ell) \tag{8}$$

Example 3.3. Returning to the example of Figure 1, consider again the window $\mathcal{W} = \{p_1, p_2, p_3, p_4\}$. Figure 4a shows the textual (cosine) distance, while Figure 4b the spatial distance for each pair of posts. For summary $S = \{p_1, p_3\}$, its textual diversity is computed as $div^T = 1 - sim^T(p_1, p_3) = 2/3$, and its spatial diversity is simply $div^S = dist(p_1, p_3) = 0.20$. □

Based on the definitions of these two quality measures of a summary, we are now ready to state our problem.

Definition 5. *For each sliding window $\mathcal{W}$ over a stream of posts, determine the summary S^* of size k that maximizes the objective function:*

$$S^* = \underset{S \subseteq \mathcal{W}, |S|=k}{\arg\max} \; f(S),$$
$$f(S) = \lambda \cdot cov(S) + (1 - \lambda) \cdot div(S)$$

where λ determines the tradeoff between coverage and diversity. □

Example 3.4. Figure 5 shows the coverage and diversity scores for each possible 2-post summary. Assuming parameter values $\alpha = 0.5$ and $\lambda = 0.5$, the objective score of each summary, depicted as the last column in the figure, is simply the average of the four individual scores. For example, summary $\{p_1, p_3\}$ has a total score of about 2.38. The best 2-post summary, however, is $\{p_2, p_4\}$ with a score of about 2.45. □

4 RELATION TO MAX-SUM DIVERSIFICATION

Our problem formulation for a single instantiation of the window is an instance of the *max-sum diversification* problem as defined in [12]. To see this, recall that coverage and diversity are defined as a summation over posts, and as summation over all pairs of posts, respectively. Thus the objective function can be rewritten as:

$$f(S) = \lambda \cdot \sum_{p \in S} cov(p) + (1 - \lambda) \cdot \sum_{\{p_i,p_j\}:p_i \neq p_j \in S} div(p_i, p_j). \tag{9}$$

It is well known that the static version of this problem can be approximated with a factor of 2 using a simple quadratic-time greedy algorithm (see [12]). In the following, we show that based on previous work one can devise an even simpler linear-time greedy algorithm that has the same approximation ratio. Although a simple observation, this is an important result which has gone undetected in the past few years.

As observed in [12], max-sum diversification is related to the *maximum dispersion* problem [15, 25]. Given a set O of objects and a distance function $d(o_i, o_j)$ between objects, determine a set S of k objects such that the sum of pairwise distances

$$g(S) = \sum_{\{o_i,o_j\}:o_i \neq o_j \in S} d(o_i, o_j) \tag{10}$$

is maximized. The most interesting case is when distance d satisfies the triangle inequality.

A 4-approximation for this problem is constructed in [25], introducing a quadratic time greedy algorithm. Initially, the two objects that have the largest distance are selected, and are inserted in the result (note that this step is responsible for the quadratic time). Then, the following is repeated $k - 1$ times: insert the object o' which maximizes the marginal gain:

$$\max_{o' \in O \setminus S} \sum_{o \in S} d(o', o). \tag{11}$$

Later, another quadratic time greedy algorithm was devised in [15], which results in a 2-approximation. As before, the algorithm starts with the two objects having the largest distance. Then, the following is repeated $k/2-1$ times: insert into the result set the two objects o_i, o_j having the largest distance $\max_{\{o_i,o_j\}:o_i \neq o_j \in O \setminus S} d(o_i, o_j)$. Note that every step of this algorithm requires quadratic time. An adaptation of this algorithm was popularized by [12] as the state-of-the-art solution for the max-sum diversification problem.

However, the maximum dispersion problem was revisited in [2], termed the *remote clique* problem. There the authors observed that the simple greedy heuristic from [25] can give a linear time 2-approximation. We call this algorithm GA — introduced as *1-Greedy Augment* in [2]. GA starts with a random object in the result set (as opposed to the two farthest objects in [25]). Then, exactly like [25], it iteratively appends to the result the object maximizing the marginal gain (Equation 11).

We finally show how GA can be adapted to our problem, similar to how [12] adapted the algorithm of [25]. The first step is to rewrite function $f(S)$ (Equation 9) so as to match function $g(S)$ (Equation 10). By setting

$$d(p_i, p_j) = \frac{2 \cdot \lambda}{k \cdot (k-1)} \cdot \big(cov(p_i) + cov(p_j)\big) + (1 - \lambda) \cdot div(p_i, p_j),$$

we can rewrite $f(S) = \sum_{\{p_i,p_j\}:p_i \neq p_j \in S} d(p_i, p_j)$.

The idea behind the transformation is easier to understand in terms of a complete graph having as nodes the posts in S, node weights equal to $\lambda \cdot cov(p)$ and edge weights equal to $(1-\lambda) \cdot div(p_i, p_j)$. Then, Equation 9 sums the weights over all nodes and all edges. However, we want to have a sum only over edge weights as in Equation 10. Thus, the transformation pushes the node weights to the edge weights, accounting for the fact that they will be included a total of $\frac{k \cdot (k-1)}{2}$ times (once for every edge adjacent to a node).

The final step for making GA applicable to our problem is to guarantee that the newly defined distance function $d(p_i, p_j)$ is non-negative and satisfies the triangle inequality. For the former, non-negativity holds because all multiplying terms are non-negative, whereas coverage and diversity are by definition also non-negative. Regarding the latter, we only need to consider whether $div(\cdot,\cdot)$ satisfies the triangle inequality, because the coverage terms disappear from the inequality and factor $1-\lambda$ is positive. While spatial distance (Euclidean or great-circle distance) is a metric, the cosine distance used in Equation 6 is not. A simple fix is to define distance in terms of the angle between vectors, instead of its cosine.[1] Then, because $div(\cdot,\cdot)$ is a non-negative linear combination of metrics, it also observes triangle inequality (the direction of a sum of inequalities is preserved).

5 ALGORITHMIC APPROACH

As previously discussed, if we consider any individual instantiation of the sliding window, our problem formulation is identical to the max-sum diversification problem. Thus, we can apply the adaptation of the greedy algorithm in [2] (discussed in Section 5.2.1) to summarize the contents of each window. However, such an approach is impractical since the sliding window can be arbitrarily large, hence storing its entire contents is not an option. So, we need to devise an efficient solution that operates on limited memory.

To achieve this, we need to address two tasks. The first (Section 5.1) is *how to compute the coverage* of posts without having the entire window's contents. Recall that the coverage of a single post is computed as the sum of its cosine similarity with each post in the window. The second task (Section 5.2) is *how to construct the summary* without having the entire window's contents. While the problem has been studied for landmark windows with limited memory [23], as well as sliding windows without memory restrictions [9], to the best of our knowledge it has not been addressed for sliding windows under limited memory.

5.1 Computing Coverage

To compute the coverage without keeping the entire window contents, we exploit the linearity of the inner product — the cosine similarity of two normalized vectors is their inner product. Note that in the following discussion, we use the term coverage to refer to both textual and spatial coverage, as they are both defined as a sum of inner products.

Our approach is based on the notion of window *pane* (or subwindow) [19]. We assume that the size of the window ω is a factor of its slide step β, e.g., a window of 24 hours sliding every one hour. The window is thus naturally divided into $m = \omega/\beta$ panes. Each time the window slides, all tuples within the oldest pane expire, while new tuples arrive in the newest pane, termed *current*.

In what follows, we denote as $\mathcal{W}$ the current and as $\mathcal{W}'$ the previous window instantiation. We also denote as $\mathcal{W}^-$ the expired pane of the previous window, and refer to the current pane as $\mathcal{W}^+$, i.e., $\mathcal{W}^- = \mathcal{W}' \setminus \mathcal{W}$ and $\mathcal{W}^+ = \mathcal{W} \setminus \mathcal{W}'$. To enumerate the panes in the window, we simply use the notation $\mathcal{W}_1$ through $\mathcal{W}_m$.

For each pane $\mathcal{W}_i$, we define its information content W_i as the vector:

$$W_i = \sum_{p \in \mathcal{W}_i} p. \tag{12}$$

It is then easy to see that the coverage of a post p can be efficiently computed using the information contents of the m panes:

$$cov(p) = \sum_{i=1}^{m} \sum_{\tau} W_i[\tau] \cdot p[\tau], \tag{13}$$

where τ represents either a keyword or a region. This implies a simple solution to computing the coverage. Instead of requiring the set of all posts within a window, it suffices to store only a few vectors, that is, the information content of each pane. Therefore, the cost of computing the coverage of a post is essentially constant. When the window slides, we throw away the information content of the expired pane $\mathcal{W}^-$, and begin aggregating posts in the current pane $\mathcal{W}^+$ to form its information content.

Example 5.1. Returning to the example of Figure 1, assume that the four posts appear at timestamps as depicted in Figure 2. The current window $\mathcal{W}$ contains all posts, while the previous window instantiation contains the first three, i.e., $\mathcal{W}' = \{p_1, p_2, p_3\}$. We next explain how the coverage can be efficiently computed as the window slides from $\mathcal{W}'$ to $\mathcal{W}$.

The window consists of 4 panes, which we enumerate from older to newer as $\mathcal{W}_1, \ldots, \mathcal{W}_4$, with $\mathcal{W}_4 \equiv \mathcal{W}^+$ being the current pane. Pane $\mathcal{W}_1$ is empty, while pane $\mathcal{W}_2$ contains post p_1 and thus its (textual and spatial) information content coincides with that of p_1: $W_1^T = (1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3}, 0, 0)$ and $W_1^S = (0, 0, 1, 0, 0, 0, 0, 0, 0)$. Pane $\mathcal{W}_3$ contains posts p_2, p_3, and thus has textual content $W_3^T = p_2^T + p_3^T = (1/\sqrt{3}, 1/\sqrt{3}, 0, 2/\sqrt{3}, 2/\sqrt{3})$, and spatial content $W_3^S = p_2^S + p_3^S = (0, 0, 0, 0, 2, 0, 0, 0, 0)$. Finally, the information content of the current pane $\mathcal{W}_4$ coincides with the single post p_4 it contains: $W_4^T = (1/\sqrt{3}, 0, 1/\sqrt{3}, 1/\sqrt{3}, 0)$ and $W_4^S = (0, 0, 1, 0, 0, 0, 0, 0, 0)$.

As the window slides, we first compute the information content of the current pane $\mathcal{W}^+$. Then, to compute the coverage of a post with respect to the window $\mathcal{W}$ after the slide, we can simply subtract the coverage due to the expired pane $\mathcal{W}^-$ and add the coverage due to the current pane $\mathcal{W}^+$. So, for example, the textual coverage of post p_3 is updated by adding $sim^T(W^+, p_3) - sim^T(W^-, p_3) = (1/\sqrt{3}, 0, 1/\sqrt{3}, 1/\sqrt{3}, 0) \cdot (0, 1/\sqrt{3}, 0, 1/\sqrt{3}, 1/\sqrt{3}) = 1/3$. □

5.2 Building the Summary

In this section, we describe several strategies for building a summary over the sliding window of posts. All approaches (except the baseline) operate without storing the entire contents of the window. They differ in what (limited) information they store across the window panes and in the way they construct the summary or

[1] We chose not to change the definition of textual diversity, because of people's greater familiarity with cosine distance.

update the previous one. In all strategies, we describe the operation necessary in a single window. That is, we assume that the information content for the current pane has been constructed as per the previous section, and thus the coverage of any post can be computed using Equation 13.

5.2.1 Baseline Strategy. The baseline strategy (BL) stores the entire contents of the window and is thus impractical, serving only as a benchmark to the quality of the summary. The method described here is the adaptation of the GA algorithm [2] discussed in Section 4. BL builds the summary incrementally, starting with an empty set. Then, at each step it inserts the post that maximizes the marginal gain of the objective function. Given a summary S, the marginal gain of a post p is:

$$\phi(p) = \lambda \cdot cov(p) + (1 - \lambda) \cdot div(p, S). \quad (14)$$

Note that GA initializes the summary with a random object because it cannot differentiate among objects when the summary is empty. On the other hand, BL can differentiate among posts, and thus it selects as initial the post that has the largest coverage.

Example 5.2. Consider the posts of Figure 1 and the window of Figure 2. We show how BL constructs a 2-post summary for the current window $\mathcal{W}$, assuming $\alpha = 0.5$, $\lambda = 0.5$. Recall that BL knows all posts in the window. The algorithm first inserts p_1 in the summary, picking it randomly among the posts as they all have the same coverage, equal to 1. Then it computes the marginal gain of each post p using Equation 14. For our purposes though, we can use the equivalent definition $\phi(p) = f(\{p_1, p\}) - f(\{p_1\}) = f(\{p_1, p\}) - 0.5$, and refer to the last column of Figure 5 that gives the values for the first term. We can see that post p_3 maximizes the marginal gain, and is thus picked by BL to construct the summary $\{p_1, p_3\}$. Notice that the greedy strategy of BL does not result in the optimal summary $\{p_2, p_4\}$; yet, the constructed summary achieves about 97% of the optimal score. □

If we assume m panes in the window, each with an equal number n of posts, we see that the memory footprint of BL is $O(k + m \cdot n)$. BL performs k passes over all posts computing for each post its diversity with respect to at most k summary posts. Thus, BL requires time $O(k^2 \cdot m \cdot n)$ to construct the summary of a window. Note that coverage is computed only once for each post at a constant cost, and thus its total cost $O(m \cdot n)$ is subsumed by that of building the summary. The running time can be improved in practice using the techniques described in Section 6.

5.2.2 Online Interchange Strategy. The work in [23] describes an online algorithm for solving the max-sum diversification problem on an ever increasing stream of objects. This approach essentially solves the problem for a landmark window, which spans from a fixed point in the past until the present. For our purposes, we adapt this algorithm to our problem involving sliding windows, where both the start and the end point of the window slide. We refer to this algorithm as OI.

OI constructs the summary S of the current window by making incremental changes to the summary S' constructed for the previous window. Initially the summary is constructed as the previous summary excluding any expired posts contained in that summary, i.e., $S \leftarrow S' \setminus \mathcal{W}^-$. Then, each newly arrived post p is examined in sequence. If the summary is not yet full ($|S| < k$), the post is simply inserted. Otherwise, the algorithm identifies the best post p^- to evict from the summary in favor of the currently examined post p. Specifically, p^- is the post maximizing the objective score of the summary $S \setminus \{p'\} \cup \{p\}$. If the eviction of p^- and the insertion of p results in an increase of the objective function, the algorithm proceeds with the replacement.

Example 5.3. Consider the posts of Figure 1 and the window of Figure 2, and assume that the summary for the previous window is $\{p_1, p_3\}$. Strategy OI will consider substituting either p_1 or p_3 for fresh post p_4 at the current pane. Specifically, it will calculate the objective scores of the two possible summaries: $\{p_1, p_4\}$ and $\{p_3, p_4\}$. From Figure 5, we can see that the latter achieves a better score of about 2.40. Moreover, this score is higher than that of the existing summary $\{p_1, p_3\}$. Hence, OI will discard p_1 and choose to keep p_4 along with p_3. Observe that OI constructs a better summary than the static algorithm because, in this example, BL makes a bad first choice by picking p_1, which cannot later undo. In contrast, OI is allowed to evict p_1, constructing a better summary. □

The OI algorithm operates on limited memory, requiring space of $O(k + n)$. For each post in the current pane, OI computes the objective score of k possible sets (one for each possible substitution of the post in the summary). Because these examined sets have significant overlap with each other, the computation of the objective score for each set can be efficiently implemented in $O(k)$ time by some clever bookkeeping [23]. Thus, the running time of OI is $O(k^2 \cdot n)$. Note that OI needs to also compute the coverage of $k + n$ posts, but since this incurs constant cost per post, its total cost contributions is subsumed. Unfortunately, OI cannot take advantage of the optimization discussed later in Section 6.

5.2.3 Oblivious Summarization. The oblivious summarization (OS) strategy, in contrast to OI, does not try to improve on the existing summary. Rather, it rebuilds the summary from scratch selecting among the posts in the current pane and those (not expired) in the previous summary. Therefore it applies the GA algorithm on the set $(S \setminus \mathcal{W}^-) \cup \mathcal{W}^+$. Naturally, the difference with BL is in the posts considered for inclusion in the summary; BL considers all window posts, whereas OS has fewer options.

Example 5.4. As in the example for the OI algorithm, assume that the previous summary is $\{p_1, p_3\}$. OS knows only about those two posts, plus the post p_4 in the current pane. Hence OS will construct a summary among those three posts using the GA algorithm. As in BL, the first pick is post p_1. Then among p_3 and p_4, the former has a higher marginal gain, which we can compute using equation $\phi(p) = f(\{p_1, p\}) - 0.5$ and the objective scores in Figure 5. Thus, OI constructs the current window summary $\{p_1, p_3\}$. □

The OS strategy requires space $O(k + n)$, and makes k passes over all n posts in the current pane. For each post, it computes its diversity with respect to at most k summary posts. Thus, the running time of OS is $O(k^2 \cdot n)$, which also subsumes the $O(k + n)$ cost for computing coverage. The OS strategy can also benefit from the optimization of Section 6.

5.2.4 Intra-Pane Summarization. The key idea of intra-pane (IP) summarization is to store a brief summary over each pane, and then use these sub-summaries to derive a summary for current window. Therefore at each window, IP constructs a local summary of size k' of the current pane using the GA algorithm. This summary is then stored along the pane unaltered until its expiration. To compute the window summary, IP once again employs the GA algorithm, but this time over the summary posts of each pane.

Example 5.5. Consider the same setting as for the OI and OS strategies. Furthermore, assume that IP creates local pane summaries by keeping a single post per pane (i.e., $k' = 1$), and let p_2 be the selected post for pane $\mathcal{W}_3 = \{p_2, p_3\}$; the local summaries for the other panes are trivial. So, IP is aware of posts p_1, p_2, p_4, and uses the GA algorithm to construct a 2-post summary among them. As before, let p_1 be the first pick. Then among p_2 and p_4, IP picks the former as it has a higher marginal gain (refer to Figure 5). Thus, IP constructs the current window summary $\{p_1, p_2\}$. □

IP needs k' space per pane, in addition to storing the contents of the current pane. Therefore it requires space $O(k' \cdot m)$. For a given window, IP invokes GA twice: once to construct the current pane's local summary with a running time of $O(k'^2 \cdot n)$, and another to construct the window summary over the pane summaries (a total of $k' \cdot m$ posts) with a cost of $O(k^2 \cdot k' \cdot m)$. This incurs a running time of $O(k'^2 \cdot n + k^2 \cdot k' \cdot m)$. In addition, IP computes the coverage (at a constant cost) of every post it stores. Since IP stores $k' \cdot (m-1)$ old posts, and n fresh posts, the total cost of computing coverage is lower than that of constructing the summary. Furthermore, IP can benefit from the optimization discussed in Section 6.

6 SPATIO-TEXTUAL OPTIMIZATIONS

The main bottleneck in all methods described above is that each post has to be evaluated individually regarding its eligibility for the summary. However, in practice, many posts may be similar to each other. In that case, it is possible to group such similar posts together and then make a decision collectively for the group, i.e., whether any post among those should be included in the summary.

In the following, we elaborate on this idea and present a process in two stages for achieving this purpose. The first stage (Section 6.1) involves partitioning the available posts into groups. The second stage (Section 6.1), given a group of posts and a summary, establishes upper and lower bounds for the coverage and the diversity of each post in the group with respect to the given summary.

6.1 Spatio-Textual Partitioning

Partitioning the posts in each pane is based on both their spatial and textual information. Given that each post belongs to exactly one region ρ, we adopt a *spatial-first* partitioning, e.g., in a uniform grid partitioning into cells or a planar tessellation into non-overlapping tiles [21]. Let $\mathcal{P}$ denote a set of posts contained within the same spatial partition. Then, the next step is to further partition $\mathcal{P}$ textually, so that the resulting subsets of posts are as homogeneous as possible with respect to the keywords they contain. The latter condition is helpful for deriving tighter bounds. Based on this, we formulate next the criterion for the textual partitioning.

Let $\Psi_{\mathcal{P}}$ denote the union of the keyword sets of the posts in $\mathcal{P}$. Assume also a partitioning Γ of $\mathcal{P}$ into the subsets $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{|\Gamma|}$. We define the *gain* $g(\mathcal{P}, \mathcal{P}_i)$ of each subset $\mathcal{P}_i$ w.r.t. $\mathcal{P}$ as the reduction rate of the size of the corresponding keyword set, i.e.:

$$g(\mathcal{P}, \mathcal{P}_i) = \frac{|\Psi_{\mathcal{P}}| - |\Psi_{\mathcal{P}_i}|}{|\Psi_{\mathcal{P}}|} \tag{15}$$

This implies that the gain is higher for partitions that have a lower number of distinct keywords. Then, the overall gain resulting from partitioning Γ is defined as:

$$g(\mathcal{P}, \Gamma) = \frac{\sum_{\mathcal{P}_i \in \Gamma} g(\mathcal{P}, \mathcal{P}_i)}{|\Gamma|} \tag{16}$$

Using this gain function, we can partition the initial set of posts recursively, applying a greedy algorithm. At each iteration, the algorithm selects one keyword for splitting, and partitions the initial set into two subsets, according to whether each post contains that keyword or not. Selecting the keyword on which to split is based on finding the keyword which results in the partitioning with the maximum gain. Then, each of the resulting subsets is partitioned recursively, until the desired number of partitions is reached or until there is no significant gain by further partitioning.

Nevertheless, performing the above check over all the candidate keywords during each iteration is time consuming. A compromise is to perform this computation offline, at a lower rate or using a subset of the stream, to identify a set of keywords that are good candidates for partitioning, and then apply these ones, updated periodically, to partition the posts in each pane. An even simpler alternative is to rely on the most frequent keywords for partitioning, since the keyword frequencies are already known for each previous pane in the window, thus requiring no additional overhead. Notice that regardless of the way the partitioning is done, the correctness of the bounds presented in the next section is not affected.

6.2 Coverage and Diversity Bounds

In what follows, we focus on a particular partition $\mathcal{P}$ of our spatio-textual partitioning and describe the necessary aggregate information we need to store and how to derive upper bounds on coverage and diversity. We abuse notation and also denote by $\mathcal{P}$ the set of posts indexed in any sub-partition below the examined.

We associate with $\mathcal{P}$ the following information:

- a vector $\mathcal{P}.p^+$, which stores at each coordinate the highest weight seen among all posts in $\mathcal{P}$;
- a vector $\mathcal{P}.p^-$, which stores at each coordinate the lowest weight seen among all posts in $\mathcal{P}$;
- the set $\mathcal{P}.\Psi$ of all keywords appearing in a post in $\mathcal{P}$.

Using this information, we next discuss how to derive the bounds.

Coverage. We firstly compute an upper bound to the possible textual coverage of a post in $\mathcal{P}$ with respect to the information content W of the window or the current pane. In other words, we seek an upper bound to

$$\max_{p \in \mathcal{P}} \sum_{\psi} W[\psi] \cdot p[\psi]. \tag{17}$$

We construct two bounds, and select the tighter one. The first trivially uses the $\mathcal{P}.p^+$ vector to upper bound a post from $\mathcal{P}$:

$$cov^T(p \in \mathcal{P})^+_I = \sum_{\psi} W[\psi] \cdot \mathcal{P}.p^+[\psi], \tag{18}$$

The second bound is based on the property that the cosine of two vectors is maximized when the vectors are parallel to each other. In our case, this translates to constructing a unit vector x parallel to W (unit because vectors in $\mathcal{P}$ are normalized). However, the inner product of W with this x would overestimate the coverage of posts in $\mathcal{P}$. As a matter of fact, vector x is constructed independently of the posts within partition $\mathcal{P}$, and thus such an upper bound trivially applies to all partitions. A tighter bound can be derived if we first project W to the dimensions corresponding to keywords in $\mathcal{P}$ and then take the unit vector parallel to W. Therefore, the second upper bound is:

$$cov^T(p \in \mathcal{P})^+_{II} = \frac{1}{\|W'\|} \cdot \sum_{\psi} W[\psi] \cdot W'[\psi] = \|W'\|, \tag{19}$$

where W' is the aforementioned projection of W, i.e.,

$$W'[\psi] = \begin{cases} W[\psi] & \text{if } \psi \in \mathcal{P}.\Psi \\ 0 & \text{otherwise.} \end{cases} \tag{20}$$

We thus have the following result.

Lemma 6.1. *The previously defined $cov^T(p \in \mathcal{P})^+_I$ and $cov^T(p \in \mathcal{P})^+_{II}$ are upper bounds to the coverage of any post p in partition $\mathcal{P}$.*

Regarding the spatial coverage, observe that all posts in the partition have the same coverage (they fall in the same region), which is computed exactly as $cov^S(p \in \mathcal{P}) = \sum_{p' \in \mathcal{W}} |\rho(p'.\ell) = \rho(\mathcal{P}.\ell)|$.

Diversity. Next, our goal is to compute an upper bound to the possible textual diversity of a post in $\mathcal{P}$ with respect to summary S, i.e., we seek an upper bound to

$$\max_{p \in \mathcal{P}} \sum_{p' \in S} \left(1 - \sum_{\psi} p'[\psi] \cdot p[\psi]\right) = |S| - \min_{p \in \mathcal{P}} \sum_{p' \in S} \sum_{\psi} p'[\psi] \cdot p[\psi].$$

Similar with the case of coverage, we can derive a diversity upper bound in two ways and employ the tighter. The first is by using the $\mathcal{P}.p^-$ vector to lower bound the inner products:

$$div^T(p \in \mathcal{P}, S)^+_I = |S| - \sum_{p' \in S} \sum_{\psi} p'[\psi] \cdot \mathcal{P}.p^-[\psi].$$

The second is again based on a geometric property of the inner product. In general, the inner product between two vectors is minimized when the vectors are parallel but in opposite directions. In the case of non-negative vectors, given a vector p' the non-negative unit vector x that maximizes their inner product must be parallel to one of the axes (intuitively in a direction as far away from p' as possible), and in particular the axis where p' has its smallest coordinate. To construct a tighter bound in our setting, we need to consider only the axes (dimensions) corresponding to keywords present in posts of $\mathcal{P}$. Therefore, the second upper bound is:

$$div^T(p \in \mathcal{P}, S)^+_{II} = |S| - \sum_{p' \in S} \min_{\psi \in \mathcal{P}.\Psi} p'[\psi].$$

Lemma 6.2. *The previously defined $div^T(p \in \mathcal{P}, S)^+_I$ and $div^T(p \in \mathcal{P}, S)^+_{II}$ are upper bounds to the diversity of any post p in partition $\mathcal{P}$ to summary S.*

Proof. The proof for the first upper bound follows from $p[\psi] \geq \mathcal{P}.p^-[\psi]$.

For the second bound, we use the following inequality:

$$\min_p \sum_i x[i] \cdot p[i] \geq \min_{i: p[i] \neq 0} x[i],$$

which holds for any unit vector p with positive coordinates.

Applying the inequality for vector $x = \sum_{p' \in S} p'[\psi]$ we get that $\sum_{p' \in S} p'[\psi] \geq \min_{\psi \in \mathcal{P}.\Psi} p'[\psi]$, where condition $\psi \in \mathcal{P}.\Psi$ is equivalent to $\psi : p[\psi] \neq 0$. The lemma follows after multiplying the resulting inequality with -1 and adding $|S|$. □

Regarding spatial diversity, we can upper bound it using the maximum possible distance between summary posts and the minimum bounding rectangle (MBR) of all posts in $\mathcal{P}$.

$$div^S(p \in \mathcal{P}, S)^+ = \sum_{p' \in S} \text{maxdist}(p', \mathcal{P}),$$

where $\text{maxdist}(p', \mathcal{P})$ returns the maximum distance of the $\mathcal{P}$'s MBR to point p'.

7 EXPERIMENTAL EVALUATION

In this section, we describe our experimental setting and we present the results of our experiments.

7.1 Experimental Setup

We present the datasets, the parameterization, and the performance criteria used to evaluate the various methods.

Datasets. We conducted tests against two real-world datasets from Flickr and Twitter. The first comprises 20 million geotagged images extracted from a publicly available dataset by Yahoo Labs and Flickr[2]. The contained images have worldwide coverage and span a time period of 4 years, from 01/01/2010 to 31/12/2013. Each image is associated with about 6 keywords on average. The second dataset comprises 20 million geotagged tweets, available online[3] and also used in [6]. It has worldwide coverage and spans a period of 9 months, from 01/04/2012 to 28/12/2012. The average number of keywords per post is 5.7.

Parameters. To compare the performance of our methods, we process both datasets in a streaming fashion, using the sliding window model, as explained in Section 3. In our experiments, we set the default pane size to $\beta = 4$ hours. We have chosen a rather large value so that the number of posts contained in the resulting panes is in the order of a few thousands, thus essentially compensating for the fact that these datasets are small samples of the actual stream of posts in these sources. Specifically, the average number of objects per pane is about 2,000 for Flickr and 12,000 for Twitter. Moreover, we set the default window size to $m = 12$ panes, and the default summary size to $k = 15$ objects.

Unless otherwise specified, the default value for both weight parameters α (Equation 2) and λ (Definition 5) is set to 0.5, thus weighting equally the spatial and textual dimensions, as well as the two criteria of coverage and diversity. For the IP method, we set the size of each intra-pane summary to $k' = 15$ objects. Finally, for the spatial partitioning used both in computing the spatial coverage

[2] https://code.flickr.net/category/geo/

[3] http://www.ntu.edu.sg/home/gaocong/datacode.htm

(Equation 5) as well as in spatio-textual partitioning (Section 6.1), we use a uniform grid with resolution 64×64 cells.

Performance measures. In our experiments, we compare all methods presented in Section 5.2, namely Baseline (BL), Online Interchange (OI), Oblivious Summarization (OS) and Intra-Pane Summarization (IP). In addition, for BL, OS and IP, we also consider their optimized versions as described in Section 6. These are designated by a plus sign (e.g., BL+).

To compare the performance of these methods, we examine two criteria. Firstly, we investigate their efficiency, which is measured as the average execution time required to update the summary every time the window slides. Secondly, we investigate the quality of the summaries they produce, by measuring their objective score (see Definition 5). More specifically, we compute this objective score for each summary a given method produces at every slide of the window, and we take their average over the entire stream. As explained in Section 4, computing the optimal summary (i.e., the one that maximizes the objective function) is practically infeasible. Thus, to compare the methods to each other, we use the objective score achieved by BL as a reference value, and we measure the scores of the rest of the methods as a ratio to that.

All algorithms were implemented in Java, and experiments were conducted on a server with 64 GB RAM and an Intel® Xeon® CPU E5-2640 v4 @ 2.40GHz processor, running Debian GNU/Linux 9.0.

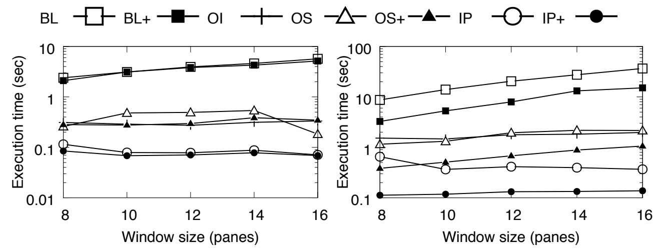
7.2 Results

Execution time. We first examine the execution time of the investigated methods. In these tests, we vary: (a) size of the window, in terms of the number m of panes it contains, Figure 6, (b) size of each pane, in terms of its duration β, Figure 7, (c) the size of each summary, in terms of the number k of objects it comprises, Figure 8 (d) tradeoff parameter α, Figure 9, and (e) tradeoff parameter λ, Figure 10. Note that logarithmic scale is used on the y axis in all plots, and that the strategy symbols are defined once in Figure 6.

A common observation is that with respect to the different strategies, BL appears to have the worst performance in most cases. This is because BL discards the previous summary and computes a new from scratch, taking into account all posts in the window. The OI and OS methods outperform BL, having a similar performance to each other. This is because OI constructs the new summary incrementally, discarding only the expired posts from the previous one, and considering only the newly arrived posts as candidates. Similarly, in OS, the benefit results from the fact that, although the summary is built from scratch, this is done by only considering the contents of the new pane and the non-expired posts in the previous summary. Yet, IP achieves an even better performance, outperforming all other methods in all experiments. In the case of IP, although all panes are considered, the candidates from the new summary are only drawn from the individual summaries of each pane, i.e., from a significantly smaller pool of posts. Due to this fact, the execution time of IP reduces significantly.

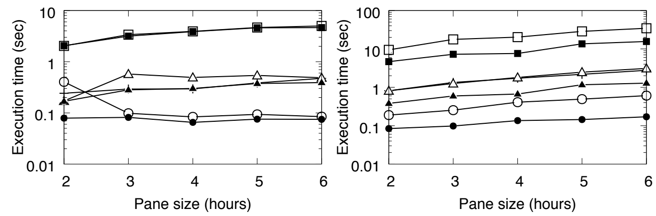
Another thing to notice is the performance of the methods with respect to their optimized versions, employing the spatio-textual partitioning and pruning presented in Section 6. The differences become more apparent in the case of Twitter, where the amount of posts per pane is about 6 times larger. In this case, the partitioning and pruning technique offers a clear benefit to all methods in which it is applicable, achieving a speedup of about 2 to 5 times.

A final remark concerns the behavior as we vary the tradeoff parameters. Specifically, when $\alpha = 0$, the textual information of



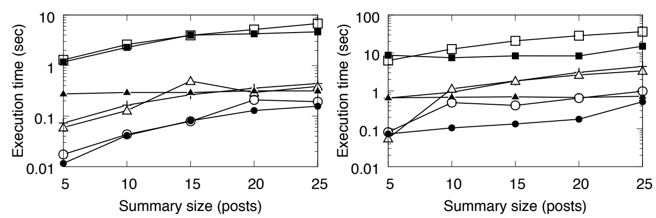
(a) Flickr (b) Twitter

Figure 6: Execution time vs. window size m.



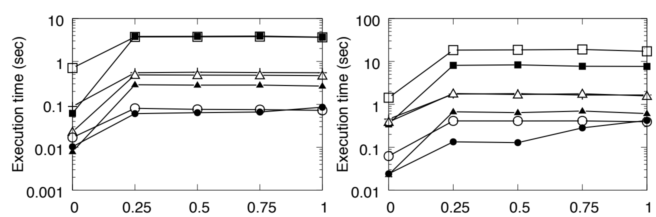
(a) Flickr (b) Twitter

Figure 7: Execution time vs. pane size β.



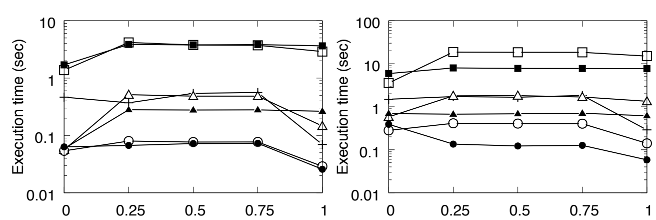
(a) Flickr (b) Twitter

Figure 8: Execution time vs. summary size k.



(a) Flickr (b) Twitter

Figure 9: Execution time vs. weight parameter α.



(a) Flickr (b) Twitter

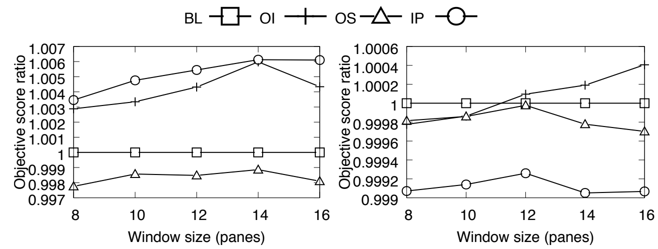
Figure 10: Execution time vs. weight parameter λ.

the posts is ignored, both when computing coverage and diversity. Figure 9 shows that in this setting all strategies become faster, with the optimized versions enjoying a larger benefit. The reason is that these optimized versions use spatial-first indices, which, in this setting, translate into spatial indices and thus are the most appropriate to have. On the other hand, the setting $\alpha = 1$ of no spatial information, does not reduce the running times, meaning that the summarization of the textual information of posts is the dominant cost. This is attributed to the fact that posts can have multiple pieces of textual content (keywords), compared to only a single piece of spatial content (location). Regarding the coverage/diversity tradeoff, note that the extreme settings $\lambda = 0$ (no coverage) and $\lambda = 1$ (no diversity), generally result in reduced running times. The effect is more pronounced for the case of no diversity, because constructing a summary that optimizes coverage alone is a simpler task; recall that the coverage of a post does not depend on the current summary and can be computed once per window.

Summary quality. Next, we investigate the objective score achieved by the different summaries computed by each method. In this set of experiments, we only consider the four different strategies without distinguishing between the optimized and non-optimized versions of each one, since the optimization applied in a strategy only affects its execution time and not the contents of the summary it produces. Moreover, as explained earlier, in each experiment we use the objective score of BL as reference, and we measure the objective scores of the rest of the methods as ratios to that. We vary the same parameters (m, β, k, α, λ) using the same defaults and along the same ranges as in the experiments regarding execution time. Results are shown in Figures 11–15.

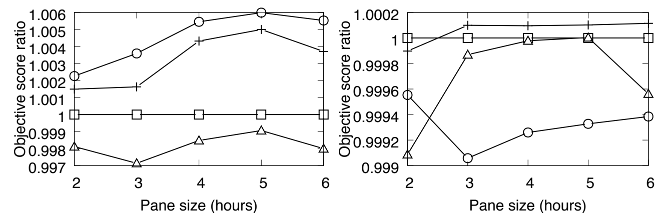
In Flickr, IP achieves the highest score, followed by OI, and both of them surpass the score of BL. However, all observed differences are rather marginal, often not exceeding 1%, except in the case of a small summary ($k = 5$ in Figure 13), where the benefits reach 3%. In fact, the situation changes in Twitter, with IP having the lowest score, whereas OI still being slightly better than BL. This noted difference for IP is attributed to the fact that the panes in the case of Twitter contain a much larger number of objects, thus relying on the intra-pane summaries to select the candidates for the new summary incurs some loss. Nevertheless, again the differences are marginal, implying that in terms of the objective score neither of these strategies appears to have a clear and significant benefit over the others. Subsequently, this leads to the conclusion that one can use the methods offering the lowest execution time without sacrificing the quality of the maintained summary over the stream. Under this reasoning, IP would be the preferred method as it is at least one order of magnitude faster than OI across all settings.

A final observation concerns behavior with respect to the tradeoff parameters. As the relative weight of textual to spatial information grows (α increases), all methods tend to produce rather similar summaries. This is because a post contains multiple pieces of textual information, which are shared across many other posts. As the emphasis on textual content increases, more posts become similar to each other, which increases the number of options available when building a summary. Thus, it becomes more likely for two different summaries to have the same quality. A last remark worth making is that when diversity is not required ($\lambda = 1$), no method can beat the static baseline BL, which in this case constructs the optimal, most representative summary. Nonetheless, all methods construct close to optimal summaries, particularly in Twitter.



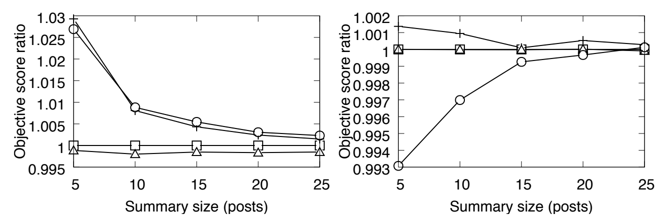
(a) Flickr (b) Twitter

Figure 11: Summary quality vs. window size m.



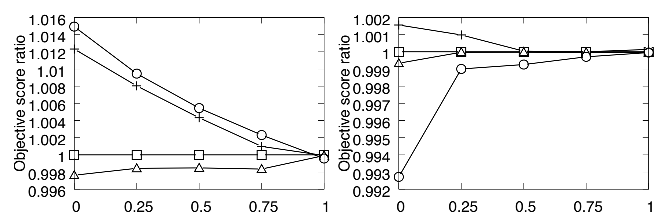
(a) Flickr (b) Twitter

Figure 12: Summary quality vs. pane size β.



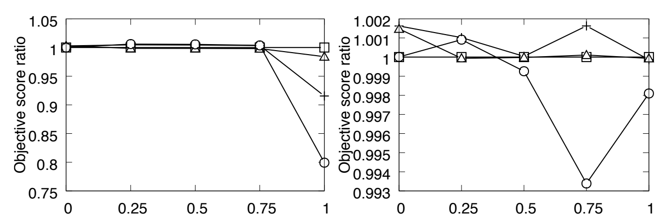
(a) Flickr (b) Twitter

Figure 13: Summary quality vs. summary size k.



(a) Flickr (b) Twitter

Figure 14: Summary quality vs. weight parameter α.



(a) Flickr (b) Twitter

Figure 15: Summary quality vs. weight parameter λ.

Comparison with other summarization approaches. Our framework offers the flexibility to obtain summaries at any point along the coverage and diversity, and the spatial and textual content dimensions. To demonstrate its effectiveness, we test two simple alternatives. The first is based on *k-means clustering*. The algorithm assigns a post p to the cluster whose centroid has the highest spatio-textual similarity with p. The cluster centroid is updated by averaging the (spatial and textual) vectors of its member posts and normalizing them. The process is repeated several times with different initial clusters, and we choose the clustering with the best total similarity. Finally, the post with the highest spatio-textual coverage (Eq. 13) is chosen from each cluster; the resulting k posts constitute the summary. This solution achieves objective scores around 40% - 45% lower than BL for both Flickr and Twitter, while being almost 100 times slower. In contrast, our strategies achieve scores at most 0.7% worse than BL and are at least 10 times faster.

The second summarization approach adapts the singular value decomposition (*SVD*)-based algorithm of [13] for the case of spatio-temporal posts. This iterative method requires linear space, i.e., storing the entire contents of the window, so it again applies to the static case. We define a post-term matrix A, where each column is the vector representation of a post, and each row corresponds to a term (keyword or spatial region). Applying SVD on matrix A yields a vector matrix $V^\intercal$, whose columns correspond to posts and represent their weighted term-frequency. At its i^{th} iteration, the algorithm inspects the i^{th} row of $V^\intercal$ to identify the post (column) that is not included in the summary and has the largest value at its i^{th} coordinate. This post is added to the summary, and the algorithm proceeds to its next iteration until $i = k$. In our tests, this approach also fared worse than our strategies, yielding objective scores nearly 22.5% and 33% lower than BL against the Flickr and Twitter datasets, respectively, while its running time was prohibitively large.

8 CONCLUSIONS

In this work, we addressed the problem of continuously maintaining a spatially and textually diversified summary over a stream of spatio-textual posts, under the sliding window model. We formally defined the problem, formulating the criteria for spatio-textual coverage and diversity over the stream of posts, and investigated different strategies that aim at minimizing the computational cost without sacrificing quality. Moreover, we proposed specific spatio-textual optimizations that further enhance the efficiency of those methods. Finally, we experimentally validated the performance of our methods using two real-world datasets from Flickr and Twitter, showing that the proposed optimizations, especially the Intra-Pane Summarization method, can achieve important performance benefits without decreasing the quality of the summary.

ACKNOWLEDGMENTS

This work was partially supported by the EU H2020 projects City.Risks (653747) and SLIPO (731581), and the NSRF 2014-2020 project HELIX (5002781).

REFERENCES

[1] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *WSDM*. 5–14.

[2] Benjamin E. Birnbaum and Kenneth J. Goldman. 2006. An Improved Analysis for a Greedy Remote-Clique Algorithm Using Factor-Revealing LPs. In *APPROX-RANDOM*. 49–60.

[3] Jaime G. Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries. In *SIGIR*. 335–336.

[4] Matteo Ceccarello, Andrea Pietracaprina, Geppino Pucci, and Eli Upfal. 2017. MapReduce and Streaming Algorithms for Diversity Maximization in Metric Spaces of Bounded Doubling Dimension. *Proc. VLDB Endow.* 10, 5 (2017), 469–480.

[5] Lisi Chen and Gao Cong. 2015. Diversity-Aware Top-k Publish/Subscribe for Text Stream. In *SIGMOD*. 347–362.

[6] Lisi Chen, Gao Cong, Xin Cao, and Kian-Lee Tan. 2015. Temporal Spatial-Keyword Top-k publish/subscribe. In *ICDE*. 255–266.

[7] Zhicheng Dou, Sha Hu, Kun Chen, Ruihua Song, and Ji-Rong Wen. 2011. Multi-dimensional search result diversification. In *WSDM*. 475–484.

[8] Marina Drosou and Evaggelia Pitoura. 2010. Search result diversification. *SIGMOD Record* 39, 1 (2010), 41–47.

[9] Marina Drosou and Evaggelia Pitoura. 2014. Diverse Set Selection Over Dynamic Data. *IEEE Trans. Knowl. Data Eng.* 26, 5 (2014), 1102–1116.

[10] Marina Drosou and Evaggelia Pitoura. 2015. Multiple Radii DisC Diversity: Result Diversification Based on Dissimilarity and Coverage. *ACM Trans. Database Syst.* 40, 1 (2015), 4:1–4:43.

[11] Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *NAACL-ANLP-AutoSum*. 40–48.

[12] Sreenivas Gollapudi and Aneesh Sharma. 2009. An axiomatic approach for result diversification. In *WWW*. 381–390.

[13] Yihong Gong and Xin Liu. 2001. Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis. In *SIGIR*. 19–25.

[14] Long Guo, Dongxiang Zhang, Guoliang Li, Kian-Lee Tan, and Zhifeng Bao. 2015. Location-Aware Pub/Sub System: When Continuous Moving Queries Meet Dynamic Event Streams. In *SIGMOD*. 843–857.

[15] Refael Hassin, Shlomi Rubinstein, and Arie Tamir. 1997. Approximation algorithms for maximum dispersion. *Oper. Res. Lett.* 21, 3 (1997), 133–137.

[16] Huiqi Hu, Yiqun Liu, Guoliang Li, Jianhua Feng, and Kian-Lee Tan. 2015. A location-aware publish/subscribe framework for parameterized spatio-textual subscriptions. In *ICDE*. 711–722.

[17] Piotr Indyk, Sepideh Mahabadi, Mohammad Mahdian, and Vahab S. Mirrokni. 2014. Composable Core-sets for Diversity and Coverage Maximization. In *PODS*. 100–108.

[18] Guoliang Li, Yang Wang, Ting Wang, and Jianhua Feng. 2013. Location-aware publish/subscribe. In *SIGKDD*. 802–810.

[19] Jin Li, David Maier, Kristin Tufte, Vassilis Papadimos, and Peter A. Tucker. 2005. Semantics and Evaluation Techniques for Window Aggregates in Data Streams. In *SIGMOD*. 311–322.

[20] Hui Lin and Jeff A. Bilmes. 2011. A Class of Submodular Functions for Document Summarization. In *ACL*. 510–520.

[21] Paul A. Longley, Mike Goodchild, David J. Maguire, and David W. Rhind. 2010. *Geographic Information Systems and Science* (3rd ed.). Wiley Publishing.

[22] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval.* Cambridge University Press.

[23] Enrico Minack, Wolf Siberski, and Wolfgang Nejdl. 2011. Incremental diversification for very large sets: a streaming-based approach. In *SIGIR*. 585–594.

[24] Kostas Patroumpas and Manolis Loukadakis. 2016. Monitoring Spatial Coverage of Trending Topics in Twitter. In *SSDBM*. 7:1–7:12.

[25] S. S. Ravi, Daniel J. Rosenkrantz, and Giri Kumar Tayi. 1994. Heuristic and Special Case Algorithms for Dispersion Problems. *Operations Research* 42, 2 (1994), 299–310.

[26] Dimitris Sacharidis, Paras Mehta, Dimitrios Skoutas, Kostas Patroumpas, and Agnès Voisard. 2017. Continuous Summarization of Streaming Spatio-Textual Posts. In *SIGSPATIAL*. 53:1–53:4.

[27] Gerard Salton and Chris Buckley. 1988. Term-Weighting Approaches in Automatic Text Retrieval. *Inf. Process. Manage.* 24, 5 (1988), 513–523.

[28] Hiroya Takamura and Manabu Okumura. 2009. Text Summarization Model Based on Maximum Coverage Problem and its Variant. In *EACL*. 781–789.

[29] Erik Vee, Utkarsh Srivastava, Jayavel Shanmugasundaram, Prashant Bhat, and Sihem Amer-Yahia. 2008. Efficient Computation of Diverse Query Results. In *ICDE*. 228–236.

[30] Marcos R. Vieira, Humberto Luiz Razente, Maria Camila Nardini Barioni, Marios Hadjieleftheriou, Divesh Srivastava, Caetano Traina, and Vassilis J. Tsotras. 2011. On query result diversification. In *ICDE*. 1163–1174.

[31] Xiang Wang, Ying Zhang, Wenjie Zhang, Xuemin Lin, and Zengfeng Huang. 2016. SKYPE: Top-k Spatial-keyword Publish/Subscribe Over Sliding Window. *PVLDB* 9, 7 (2016), 588–599.

[32] Wenjian Xu and Chi-Yin Chow. 2016. A Location- and Diversity-Aware News Feed System for Mobile Users. *IEEE Trans. Services Computing* 9, 6 (2016), 846–861.