To appear in *Proceedings of the 39th Conference on Very Large Data Bases*, August 26-30, 2013, Riva del Garda, Trento, Italy.

PhotoStand: A Map Query Interface for a Database of News Photos[*]

Hanan Samet Marco D. Adelfio Brendan C. Fruin
Michael D. Lieberman Jagan Sankaranarayanan
Center for Automation Research, Institute for Advanced Studies,
Department of Computer Science, University of Maryland
College Park, MD 20742 USA
{hjs, marco, brendan, codepoet, jagan}@cs.umd.edu

ABSTRACT

PhotoStand enables the use of a map query interface to retrieve news photos associated with news articles that are in turn associated with the principal locations that they mention collected as a result of monitoring the output of over 10,000 RSS news feeds, made available within minutes of publication, and stored in a PostgreSQL database. The news photos are ranked according to their relevance to the clusters of news articles associated with locations at which they are displayed. This work differs from traditional work in this field as the associated locations and topics (by virtue of the cluster with which the articles containing the news photos are associated) are generated automatically without any human intervention such as tagging, and that photos are retrieved by location instead of just by keyword as is the case for many existing systems. In addition, the clusters provide a filtering step for detecting near-duplicate news photos.

1. INTRODUCTION

A demo is presented of PhotoStand (see also the related NewsStand [9, 17, 21, 29], TwitterStand [6, 24], and STEWARD [12] systems) which is an example application of a general framework we are developing for retrieving multimedia data (e.g., text, images, videos) using a map query interface from a database of news articles, photos, and videos (i.e., by location in real-time which differentiates it from Google where static photos are retrieved which rely on human geotagging). The photos are associated with news articles [23] collected by monitoring the output of over 10,000 RSS news feeds and made available for map-based retrieval within minutes of publication. These feeds are processed by the NewsStand system which constantly polls them, downloads the new articles that they contain, performs a variety of tasks on them, and stores the results in a PostgreSQL database. This is motivated by our prior work on indexing spatial and temporal data [4, 5, 18–20] and similarity searching in the serial domain [16, 22, 25], as well as in a distributed domain [28].

The three major processing modules of NewsStand are its *cleaner* module, which extracts the text, images, and videos, as well as discards irrelevant objects in the feed; its *geotagger* [7, 8, 10, 11, 14], which extracts locations mentioned in the articles, enabling them to be accessed by spatial queries such as windowing or simple point location; and its *clusterer* [30], which groups articles about the same topic. A key to the NewsStand database system is its pipe server which coordinates its processing modules by assigning batches of articles to them. NewsStand's user interface enables the retrieval of clusters of news articles for display using its map user interface by executing what we term *top-k window queries*. At present, NewsStand handles about 50K articles per day and has a large underlying database of articles currently containing about 300GB of data.

The PhotoStand and TweetPhoto [3] demos are related in the sense that PhotoStand uses photos from news articles in NewsStand, while TweetPhoto uses photos from news tweets in TwitterStand [24]. In addition, the PhotoStand demo demonstrates the database querying capability of NewsStand as well as its capability to do similarity searching for news photos where the first step in the similarity detection process is based on the text associated with the photos, while the second step involves use of the actual image features (e.g., texture, color) to enable detecting near duplicates, thereby avoiding the combinatorial complexity of comparing every photo with every other photo.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 indicates how news articles (and consequently news photos) are clustered, as well as how captions are identified and extracted. Section 4 describes near-duplicate image detection. Section 5 presents the demo scenario and some underlying interaction with the database, while concluding remarks are given in Section 6.

2. RELATED WORK

Most of the work in associating geographic locations with images has dealt with images that correspond to photos and involves tags generated by humans (often just the dateline of the associated article in the case of news photos) or by a GPS device built into the camera (e.g., Flickr images). Unfortunately, user generated tags are not always sufficient to generate precise latitude-longitude coordinate values meaning that they require additional human intervention to identify the location although gazetteers do help. We limit ourselves to photos that accompany news articles and use the vector space model of the contents of article documents to help us find similar photos. These feature vectors are often sufficient to describe both the documents and the images that they contain. In fact, the feature vectors can be viewed as tags, although no humans are involved in their creation.

The idea of geotagging images based on image tags (or captions) or using the image's GPS coordinates has been extensively explored [1,2,13,26,31], often using a collection of images uploaded to Flickr. Serdyukov et al. [26] rely solely on Flickr image tags to geotag images and to place the geotagged images on a map at varying levels of granularity. While similar to PhotoStand in its representation and extraction of location information, this method of geotagging may return ambiguous

[*]This work was supported by the NSF under Grants IIS-09-48548, IIS-10-18475 , and IIS-12-19023.

results when the image tag lacks precision. Crandall et al. [2] propose an improvement over a tag-only grouping system by incorporating the GPS coordinates. This approach is not feasible for a system such as PhotoStand, which queries thousands of news sources that may not geotag their images.

The Google Maps' photos layer is another approach where a world map is overlaid with user submitted images at the location where they were taken. Similar to PhotoStand's user interface, the displayed photos change as the viewing window changes. However, the photos layer of Google Maps requires image locations to be tagged for each photo while PhotoStand automatically geotags images using the article where they were found. PhotoStand also focuses on images pertaining to news which often results in the map's photos being updated whereas photos in Google Maps are of geographical landmarks which update less often. Another example is the image hosting site Flickr that has a map that displays markers for the top geotagged locations in the world. Unlike PhotoStand, Flickr's map does not update to display new images when the viewing window changes meaning that locations having images, but not in the top geotagged image locations, are not shown.

3. IMAGE EXTRACTION

3.1 Feature Extraction

We use the *vector space model* [15] of documents, often used in text mining and information retrieval. This model represents a text document as a *term feature vector* in a d-dimensional space, where d is the number of distinct terms in every document in a corpus.

Upon receiving a new article to be clustered, we extract the article's term feature vector by computing the well-known *Term Frequency-Inverse Document Frequency* (TF-IDF) [15] score for each term in the article. This score emphasizes those terms that are frequent in a particular document and infrequent in a large corpus D of documents.

3.2 Online Clustering

We cluster using a variant of leader-follower clustering [30] that permits online clustering in both the term vector space and the temporal dimension. For each cluster, maintain a *term centroid* and *time centroid*, corresponding to the means of all term feature vectors and publication times of articles in the cluster, respectively. To cluster a new article a, check whether a cluster exists where the distance from its term and time centroids to a is less than a fixed cutoff distance ϵ. If one or more candidate clusters exist, add a to the closest such cluster, and update the cluster's centroids. Otherwise, create a new cluster containing only a.

We use a variant of the *cosine similarity measure* [27] for computing term distances between the new article and candidate clusters. The term cosine similarity measure for a article a and cluster c is defined as

$$\delta(a,c) = \frac{\overrightarrow{TFV_a} \bullet \overrightarrow{TFV_c}}{||\overrightarrow{TFV_a}||\,||\overrightarrow{TFV_c}||}$$

where $\overrightarrow{TFV_k}$ is the term feature vector of k.

In order to account for the temporal dimension in clustering, we apply a Gaussian attenuator on the cosine distance that favors those clusters whose time centroids are close to the article's publication time. In particular, the Gaussian parameter takes into account the difference in days between the cluster's time centroid and the new article's publication time. Our modified distance formula is

$$\dot{\delta}(a,c) = \delta(a,c) \cdot e^{\frac{-(T_a - T_c)^2}{2(2.2)^2}}$$

where T_a is a's publication time and T_c is c's time centroid.

To improve performance, we store cluster centroids in an inverted index that contains, for every term t, pointers to all clusters that have non-zero values for t. We use this index to reduce the number of distance computations required for clustering. When a new article a is clustered, we compute the distances only to those clusters that have non-zero values in the non-zero terms of a. As a further optimization, we maintain a list of *active* clusters whose centroids are less than a few days old. Only those clusters in the active list are considered as candidates to which a new article may be added. We remove clusters from the active list after several days, since the values from our distance function will be negligible. Together, these optimizations allow our algorithm to minimize the number of distance computations necessary for clustering articles.

3.3 Cluster Feature

The identification of entities (i.e., people, location, and organization) in news articles is facilitated by the use of a Natural Language Processing (NLP) method known as *Named-Entity Recognition* (NER) (e.g., LingPipe). Our NER tagger takes a news article as input and annotates words and phrases in the news articles of location, people and organization. Now, we can aggregate on all the articles belonging to a cluster to obtain the most common location, people, and organization names mentioned in all the articles belonging to the cluster. In other words, if an entity appears in a majority of the articles belonging to a cluster, then we can assume that it is important to the news topic. By requiring that an entity appears in most of the articles associated with a cluster, we can average out most of the entities that are noise resulting in good quality output. Note here that the feature vector corresponding to the cluster centroid, caption (described below) and the entities are all associated with each image that is extracted from an article.

3.4 Image Extraction

Now that an article has been clustered, we can extract all the relevant images from it using the feature vector of the cluster. Image extraction from a web page requires processing of the HTML tags so that a caption can be associated with each of the images in the news article. Observe that we may not be able to identify a caption for each of the images in a news article, but from our experience, we are able to do so for a large percentage of them, although some may correspond to ads or other irrelevant objects. Even though the captions of images are usually not very descriptive due to their succinctness, they still capture their content in the sense that they have terms in common with the text, and hence the captions and text are said to be *similar*.

We examine every image in the HTML page. If we can visualize the HTML as a tree structure, and the image as a node in the tree, then the idea is to look at the children nodes and a few ancestor nodes to try to collect enough text which would serve as the caption of the image. Sometimes the image may have a TITLE field associated with it, in which case it forms the caption of the image. In some cases there may be an ALT field, which can also serve as a caption. We also look for configurations where the image is embedded in a DIV element, in which case we use any text found within the DIV element. Our algorithms use several configurations such as nested DIV and TABLE structures, or combinations of them. Note that the caption is usually not very long, which means that we can simply discard any text if it is too long.

Once we have a caption for the image, we try to match the terms (i.e., words) it contains with the cluster's term centroid to see how many keywords from the cluster are found in the caption. For example, if the feature vector of the document contains "Obama", "Bohner", "Debt", and "Congress", then we would expect that one or more of these "features" be present in the caption text. Otherwise, we discard the image. Once we extracted the image, the database records the caption text and the cluster term centroid.

4. NEAR-DUPLICATE IMAGE DETECTION

In order to efficiently store and process large collections of candidate near-duplicate images we need a representation for each image which is extremely compressed and efficient to compute. It is crucial that the representation be resistant to changes in scale, saturation, hue, contrast, compression, and cropping. A hierarchical color histogram is used to follow the pattern of an image pyramid with three levels.

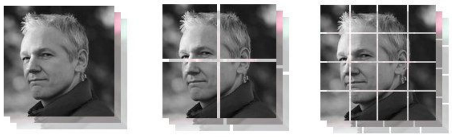


Figure 1: Illustration of the pyramid structure and color channel segments which are used to compute the hierarchical color histograms.

Thus, we utilize a global histogram from the first level, a histogram for each quadrant from the second level, and sixteen histograms from the third level. This structure allows us to encode and compare global and local features of the images, preserving spatial information. We use less memory to encode the histograms at lower levels, ensuring that the information from higher levels receives appropriate weight.

To encode an image's color information, we first convert images into the Lab color space since it approximates human visual perception. After a histogram is calculated for a channel of a segment in the image pyramid, the bins of the histogram are shifted to minimize its vector representation. This process addresses changes in hue found in news images. For images in grayscale, we only encode their lightness information, so when comparing vectors of a grayscale and color images we only consider the lightness components.

The histograms are concatenated in a feature vector, and depending on the precision needed for the application, the number of histogram bins and data used to represent them can be modified. We store each bin as a single byte, and our entire feature vector is 512 bytes. The similarity between two images can be computed as the euclidean distance between their vectors. This allows us to group or retrieve duplicates using common clustering or retrieval techniques.

Figure 2 shows examples of near-duplicate images detected by our approach. Note that most news images on the web undergo limited transformations or alterations. In particular, our method is not robust to occlusions or significant cropping. These transformations can dramatically affect the intensity and color layout of the image. In Figure 3, the near duplicate images all share similar color and intensity structure despite other differences in the images, and thus detected. But if an image is cropped so that a primary color element is removed, then the histogram is fundamentally changed and the near duplicate may not be detected.

5. DEMO SCENARIO

The PhotoStand user interface consists of a map where photo thumbnails are displayed at the locations that are deemed most appropriate to the clusters that contain the news articles with which the photos are associated (e.g., Figure 4a). Initially, the map contains photo thumbnails at locations corresponding to those deemed most relevant in the k most representative clusters, where "representative" takes into account the importance of the cluster's subject as measured by factors such as currency, size and rate of growth of the cluster in terms of velocity and acceleration rates, as well as a desire to have a good spatial distribution in the area being displayed (i.e., the viewing window). A slider is centered above the map whose movement to the right (left) allows the maximum number of different clusters for which news photos are displayed at their representative locations to increase (decrease).

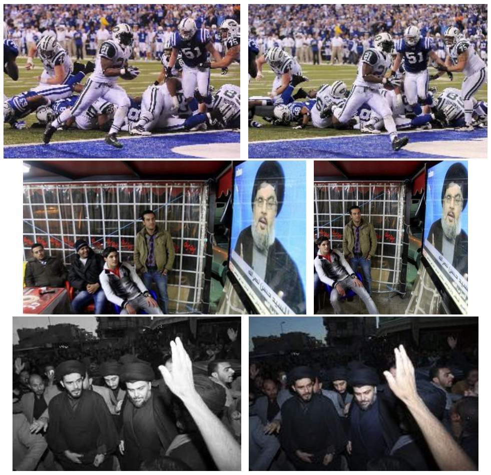


Figure 2: Examples of near-duplicate images detected in real news sources. First Row: Similar time instances. Second Row: Different image croppings. Third Row: Similar grayscale and color images.

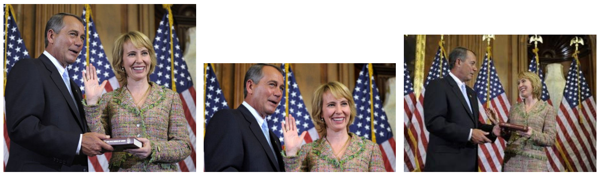
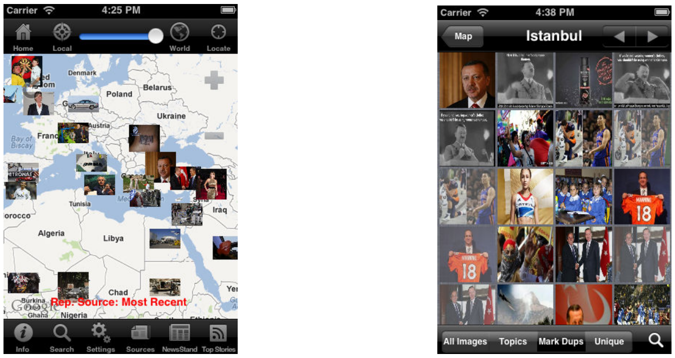


Figure 3: Three examples of near-duplicate images automatically detected from unique news sources by our system.

Executing a search for keyword w yields a set of k photo thumbnails whose captions contain w displayed at the locations that are deemed most appropriate to the clusters that contain the articles with which the photos are associated. An alternative, which we do not use at present, bases the selection of the photos on the text of the associated articles rather than the captions. The k photo thumbnails are ordered according to the importance of the clusters that contain the articles with which the photos are associated. Once a search has been initiated, all subsequent searches are restricted to the keyword. However, the searches are also restricted to the displayed part of the map (in other words, they are spatially restricted and are analogous to a spatial join operation). Users also have the option to restrict the sources of the articles with which the images are associated, as well as the language in which they are written. This can be done by specifying the names of the sources (e.g., "Washington Post"), the geographic regions in which they are published (e.g., Ireland, UK), or the language in which they are written (e.g., "French").

In scoring an image, we are looking to find the similarities between the set of location keywords or cluster keywords and the set of the image's caption words. A traditional approach for finding the similarities between sets A and B is to use the Jaccard Index which is defined as: $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$. Hence, the resultant index score ranges from 0 to 1, where 0 indicates that the two sets have nothing in common while 1 indicates the sets are the same. Unfortunately, using this approach, the frequencies of the keyword terms do not enter into play as the rank is really just based on the number of keywords that the caption has in common with the location terms or cluster terms. Using a variation of the Jaccard Index, we define the numerator to be the sum of the cardinality of the intersection of the set comprising the union of the terms associated with the



(a) (b)

Figure 4: (a) Map view showing news photos and the principal locations of their corresponding clusters (b) Grid of news photos associated with a location in ranked order where near duplicates are shown in gray

locations or news cluster and the set of words occurring in the caption and the frequencies associated with each of the words in the intersecting set. This quantity is divided by the sum of the cardinality of the union of the terms associated with a location or news cluster and the words in the caption and the total frequencies of all the terms to get an *image_score*. In order to also rank the image using its recency, the final score is defined as $(image\_score * time\_factor) + days\_offset$ where $time\_factor$ is a constant that gives appropriate weight to the score found using the modified Jaccard Index, and *days_offset* is the number of days since the module was started. This score update guarantees both that recent images will have a higher score, and that an image scoring highly using our modified Jaccard Index will not always remain the highest scoring image for a given location or cluster.

The above technique enables us to rank the photos thereby enabling us to choose the most representative photo to be displayed on the map. Users can also display the remaining photos in decreasing order of their ranking. This is achieved by the following two-step process. A single click on the photo thumbnail on the map reveals the name of the location and a small text string corresponding to a partial caption. A subsequent click on the rightward pointing arrow reveals a grid with other photo thumbnails associated with the location ranked in decreasing order of cluster relevance, which is based in part on the number of distinct news sources and several other factors. In the grid, one can mark near duplicate images as well as remove them (e.g., Figure 4b). Clicking on a photo enlarges it to take up a large part of the display screen and shows its caption. Double clicking yields the text of the associated article.

Users can also view the photos associated with a location by their associated clusters (one photo per cluster), ranked by just using cluster terms for the specific cluster and only using their frequencies in the articles in the cluster. Clusters are ranked in importance the same way as in NewsStand.

6. CONCLUSION

Both app and Web (`http://photostand.umiacs.umd.edu`) versions of Photostand exist, and a screencast demo can be seen at `http://photostand.umiacs.umd.edu/demo`.

7. REFERENCES

[1] S. Ahern, M. Naaman, R. Nair, and J. Yang. World explorer: Visualizing aggregate data from unstructured text in geo-referenced collections. In *JCDL*, pp. 1–10, 2007.

[2] D. J. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg. Mapping the world's photos. In *WWW*, pp. 761–770, 2009.

[3] B. C. Fruin, H. Samet, and J. Sankaranarayanan. Tweetphoto: photos from news tweets. In *GIS*, pp. 582–585, 2012.

[4] G. R. Hjaltason and H. Samet. Speeding up construction of PMR quadtree-based spatial indexes. *VLDBJ*, 11(2):109–137, 2002.

[5] G. S. Iwerks, H. Samet, and K. Smith. Maintenance of spatial semijoin queries on moving points. In *VLDB*, pp. 828–839, 2004.

[6] A. Jackoway, H. Samet, and J. Sankaranarayanan. Identification of live news events using Twitter. In *LBSN*, pp. 25–32, 2011.

[7] M. D. Lieberman and H. Samet. Multifaceted toponym recognition for streaming news. In *SIGIR*, pp. 843–852, 2011.

[8] M. D. Lieberman and H. Samet. Adaptive context features for toponym resolution in streaming news. In *SIGIR*, pp. 731–740, 2012.

[9] M. D. Lieberman and H. Samet. Supporting rapid processing and interactive map-based exploration of streaming news. In *GIS*, pp. 179–188, 2012.

[10] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging with local lexicons to build indexes for textually-specified spatial data. In *ICDE*, pp. 201–212, 2010.

[11] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging: Using proximity, sibling, and prominence clues to understand comma groups. In *GIR*, 2010.

[12] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. STEWARD: architecture of a spatio-textual search engine. In *GIS*, pp. 186–193, 2007.

[13] S. Overell, B. Sigurbjörnsson, and R. van Zwol. Classifying tags using open content resources. In *WSDM*, pp. 64–73, 2009.

[14] G. Quercini, H. Samet, J. Sankaranarayanan, and M. D. Lieberman. Determining the spatial reader scopes of news sources using local lexicons. In *GIS*, pp. 43–52, 2010.

[15] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *IPM*, 24(5):513–523, 1988.

[16] H. Samet. K-nearest neighbor finding using MaxNearestDist. *IEEE TPAMI*, 30(2):243–252, 2008.

[17] H. Samet, M. D. Adelfio, B. C. Fruin, M. D. Lieberman, and B. E. Teitler. Porting a web-based mapping application to a smartphone app. In *GIS*, pp. 525–528, 2011.

[18] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin. Use of the SAND spatial browser for digital government applications. *CACM*, 46(1):63–66, 2003.

[19] H. Samet, A. Rosenfeld, C. A. Shaffer, and R. E. Webber. A geographic information system using quadtrees. *Pattern Recognition*, 17(6):647–656, 1984.

[20] H. Samet and M. Tamminen. Bintrees, CSG trees, and time. *Computer Graphics*, 19(3):121–130, 1985.

[21] H. Samet, B. E. Teitler, M. D. Adelfio, and M. D. Lieberman. Adapting a map query interface for a gesturing touch screen interface. In *WWW (Companion Volume)*, pp. 257–260, 2011.

[22] J. Sankaranarayanan, H. Alborzi, and H. Samet. Efficient query processing on spatial networks. In *GIS*, pp. 200–209, 2005.

[23] J. Sankaranarayanan and H. Samet. Images in news. In *ICPR*, pp. 3240–3243, 2010.

[24] J. Sankaranarayanan, H. Samet, B. Teitler, M. D. Lieberman, and J. Sperling. TwitterStand: News in tweets. In *GIS*, pp. 42–51, 2009.

[25] J. Sankaranarayanan, H. Samet, and A. Varshney. A fast all nearest neighbor algorithm for applications involving large point-clouds. *Computers & Graphics*, 31(2):157–174, 2007.

[26] P. Serdyukov, V. Murdock, and R. van Zwol. Placing Flickr photos on a map. In *SIGIR*, pp. 484–491, 2009.

[27] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, pp. 1–20, 2000.

[28] E. Tanin, A. Harwood, and H. Samet. A distributed quadtree index for peer-to-peer settings. In *ICDE*, pp. 254–255, 2005.

[29] B. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. NewsStand: A new view on news. In *GIS*, pp. 144–153, 2008.

[30] B. E. Teitler, J. Sankaranarayanan, H. Samet, and M. D. Adelfio. Online document clustering using GPUs. In *ADBIS Workshop GID*, 2013.

[31] K. Q. Weinberger, M. Slaney, and R. van Zwol. Resolving tag ambiguity. In *ACM Multimedia*, pp. 111–120, 2008.