

# Geo-Clustering of Images With Missing GeoTags

Vishwakarma Singh  
Department of Computer Science,  
University of California,  
Santa Barbara, USA  
vsingh@cs.ucsb.edu

Sharath Venkatesha  
Department of Computer Science,  
University of California,  
Santa Barbara, USA  
sharath20284@gmail.com

Ambuj K. Singh  
Department of Computer Science,  
University of California,  
Santa Barbara, USA  
ambuj@cs.ucsb.edu

**Abstract**—Images with GPS coordinates are a rich source of information about a geographic location. Innovative user services and applications are being built using geotagged images taken from community contributed repositories like Flickr. Only a small subset of the images in these repositories is geotagged, limiting their exploration and effective utilization. We propose to use optional meta-data along with image content to geo-cluster all the images in a partly geotagged dataset. We formulate the problem as a graph clustering problem where edge weights are vectors of incomparable components. We develop probabilistic approaches to fuse the components into a single measure and then, discover clusters using an existing random walk method. Our empirical results strongly show that meta-data can be successfully exploited and merged together to achieve geo clustering of images missing geotags.

## I. INTRODUCTION AND MOTIVATION

The recent decade has witnessed an unprecedented growth in public contributed image datasets like Flickr<sup>1</sup>. Flickr images along with their meta-data like timestamps (camera time), geotags (latitude and longitude), text tags (descriptive keywords), and description are a rich source of information. Invention of new tools that can harness these rapidly growing repositories of images for interesting information will open new venues for research and opportunities for business.

Images having the latitude and longitude information (geotag) are called geotagged images. These images have especially attracted the attention of the research community because of their huge application potential [1], [2], [3], [4], [5], [6], [7]. Limitation of these works is the lack of large and useful geotagged datasets. There were more than 3 billion images<sup>2</sup> in Flickr as of November 3, 2008, but only 40 million [4] were geotagged. Crandall et al. [5] report that only 55% of the images in their collection had accurate geo tags. Our dataset has only 29% geotagged images. Methods which can automatically cluster images by geo location (geo-clustering), even in the absence or partial availability of geotags, will dramatically boost our capability to churn information from these highly under utilized repositories. Clustering will also directly help predict the location of images missing geotags. High usability of geotagged images and the fact that the majority of the images are not geotagged provide the strongest motivation to our work.

*Our goal is to develop a method to geo-cluster all the images in a dataset even if most of the images are missing their geotags.* State of the art clustering approaches like k-means or mean-shift would require geotags of all the images and therefore, cannot be directly adopted for our task. We call geotag as *primary attribute* for the task of geo-clustering. We use image visual content, image timestamp, and text tags, all together called *secondary attributes*, to supplement partly available geotags to geo-cluster all the images. We determine a weighted relationship between a pair of images for each of the attributes. We assume that a strong relationship (small distance) between a pair of images on a attribute is a strong indication of the images being geo-located.

We discuss in Section III that each of the attributes alone is insufficient for our task. Therefore, we use all the attributes in conjunction. We construct a graph for each attribute where images are the nodes and edge weights are derived from the corresponding attribute. We overlay all these component graphs to form a combined graph where an edge weight is a vector whose components are the weights obtained from different attributes. We pose our problem as clustering of nodes in this overlaid graph in Section IV. Graph clustering problems in literature have only considered scalar weights on the edges. To cluster our heterogeneous graph, we take the approach of merging the components of an edge vector to obtain an aggregated scalar weight for each edge which is discussed in Section V. This extends the reusability of existing graph clustering techniques to our problem directly as discussed in Section VI. There is no obvious way to merge the components because weights obtained from different heterogeneous sources are not directly comparable (e.g. differences in timestamps of two images cannot be directly compared against their visual similarity). We develop probabilistic methods to transform weights of edges of each graph into a space where transformed values across attributes are comparable. Then, we aggregate these values with simple functions. We also develop a logistic regression based method to fuse all the components directly to obtain a probabilistic weight for each edge of the overlaid graph. We describe our dataset and graph construction approaches in Section VII. We perform experiments to empirically verify our methods in Section VIII. Our probabilistic approaches of integrating information from different domains achieve impressive results for geo-clustering of images.

<sup>1</sup>[www.flickr.com](http://www.flickr.com)

<sup>2</sup><http://www.techcrunch.com/2008/11/03/three-billion-photos-at-flickr/>

Our major contributions are: 1) a new clustering problem for the case of missing information specifically geo-clustering of images in the case of missing geotags, 2) abstraction of the problem to a general graph clustering task, and 3) probabilistic solutions to integrate information from various heterogeneous secondary attributes.

## II. RELATED WORK

Crandall et al. [5] clustered geotagged images to find the most photographed location at different scales. They studied the effectiveness of text tags and visual content for predicting location of an image using SVM classifier. Hays et al. [2] built a training set of 6 million geotagged images after filtering on tag using geographic keyword dictionary and manually chosen words. They predicted the location of an image by clustering top-120 visually similar images from the dataset. They classified the image to the cluster which had maximum cardinality. Kalogerakis et al. [8] use a Hidden Markov Model variant to geo-locate a sequence of timestamped photographs taken by a single user using both image and temporal information.

One third of the proteins of a typical proteome lack functional annotations. Researchers have used heterogeneous data sources like genetic interactions, co-expression knowledge, and physical interaction maps to predict protein function. Sharan et al. [9] surveyed different methodologies. Methods for integrating multiple information sources for prediction are SVM classifier [10], Boltzman machine [11] and Markov Random Field [12]. All these methods are designed on per functional class basis and need to be repeated for each class separately. These methods cannot scale for the location prediction because number of geo locations can be many orders higher than protein functional classes, depending on the scale of spatial resolution. Also, a protein can have multiple functional classes, unlike geo location. Lee et al. [13] integrated information from various sources over all functional classes in a single framework using Kernel based logistic regression. The weakness of this method is large number of unknown parameters:  $((n-1)*m*(n+1))$  for  $n$  classes and  $m$  sources. Further, all these methods are tightly integrated to the task of prediction and have been tested only on smaller graphs and small set of functions.

## III. ATTRIBUTE DESCRIPTION

In this section, we describe all the four attributes and their strengths and weaknesses.

**Geotag:** Geotag is an optional data provided by a user or obtained by dragging an image on a map provided by an application. Since these manual methods are prone to errors, images of the same scene may not have the same GPS coordinate but are expected to have relatively small geodesic distance between them. Geotag is available only for a small subset of the dataset and is clearly not enough to geo-cluster all the images.

**Text Tags:** Text tags are a list of optional descriptive keywords provided by a user for an image. We use the word “tags” synonymously with text tags in this paper from now on. Tag

similarity can be used to infer the geo-collocation of two images. Text tags are optional attributes and therefore many images have missing tags. 9% of the images in our dataset have no text tags. Also, a strong tag based relationship does not always imply spatial proximity. Tags are a mix of following three broad subparts: 1) Geographic keywords (e.g. city name), 2) Image content description tags (e.g. building, or beach), and 3) Noise words like camera description. Geographic keywords are the most helpful tags but suffer from ambiguity (e.g. Mississippi river/state). These manually-defined tags are often vague, misspelt or incorrect [14], requiring sophisticated methods to filter them correctly. Image content descriptive tags can be used across images of different geo-location. These common noun terms help distinguish between content of different images but are not good predictors of geo location. Text tags cannot alone provide satisfactory geo-clustering of images because of these reasons.

**Timestamp:** Digital cameras timestamp images using their clock. Images taken in close temporal proximity by a user are highly likely to be of the same place even though they may not have similarity based on tags or content. Difference in timestamps (temporal relationship) of images taken by a user can reveal strong geo-collocation. Temporal relationship has already been successfully used to group images of a single user based on events [15], [16]. Temporal relationship cannot be defined between images of two different users. Therefore, timestamp cannot group images of the same scene taken by two different users and hence, cannot perform our task alone.

**Image Content:** High visual content similarity between images indicates that the images are of the scene and can be used to infer geo-collocation of images. This property holds true for landmark images (e.g. Tajmahal, Pisa Tower) but may not be true for all kinds of natural images (e.g. beach). Images of beaches taken by two different users at two different locations may have high visual similarity. Further, visual similarity is computed using low level features of images like edge, shape, pixel intensity gradient and color. The best reported average precision for similarity search in landmark images is 64.5% [17] using state of the art SIFT [18] features. Geometric verifications [18], [6], [17] have also been used to filter out false matches but only with limited success. Clustering images just based on visual similarity will yield unsatisfactory results in terms of geo location because of these issues. Our empirical results concur with the findings of Crandell et al. [5] that visual similarity alone is a poor indicator of geo location.

## IV. PROBLEM FORMALIZATION

Here, we formalize our task as a general graph clustering problem. We define a component graph  $G_k$  for each attribute  $k$ .  $V(G_k)$  are the nodes of the graph corresponding to images. Weights of the edges are obtained from respective attributes. Geotag yields geo graph  $G_1$  where an edge weight  $l$  is geodesic distance between a pair of images. Timestamp gives temporal graph  $G_2$  where an edge weight  $t$  is difference in timestamp of a pair of images. Text tag gives tag graph  $G_3$  whose edge has weight  $d$  equal to the complement of the

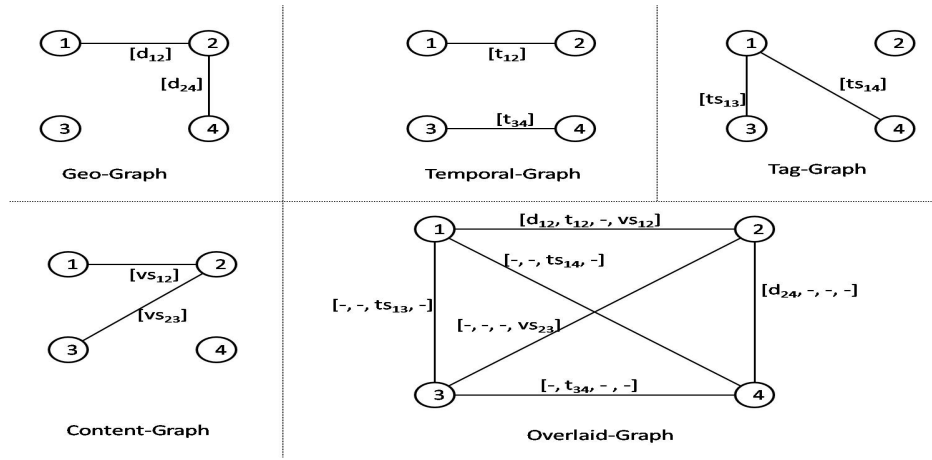


Fig. 1. This figure shows each of the component graphs with edges. Missing edges in geo graph and tag graph is because of missing geotags and text tags respectively. Edges are not defined between images of different users in temporal graph. Content graph has missing edges for zero image similarity. All the graphs are merged into overlaid graph. Edge vector of the overlaid graph has weights from components graphs. Missing weights in a vector are represented by ‘-’.

tag similarity between images. Visual content graph  $G_4$  has edge weight equal to the complement of the visual similarity between images. Edge weights in all the graphs are representative of distances between images. We overlay these graphs to build a graph  $G'$  whose edge weight is given by a vector  $W = [l, t, d, v]$ . An overlaid graph is shown in Figure 1. We formalize our problem as: *Given a graph  $G'$  with a constraint vector  $W$  on each edge having incomparable components, cluster the nodes with respect to a missing or partly available attribute.*

## V. ALGORITHMS FOR DATA FUSION

In this section, we develop methods for clustering the images using multiple attributes. Clustering of nodes of a graph with vector weighted edges is challenging. Further, components of the weight vector  $W$  are of conceptually different space and scale and therefore, directly incomparable. Temporal differences measured in seconds cannot be merged with geodesic distances measured in meters. We take the approach of transforming the components of each type into a comparable probability space. Then, we merge the transformed components of an edge vector to get a scalar value.

A naive approach of merging different components of  $W$  can be Thresholding. We decide a threshold for each component type. A component is assigned a value of 1 if it's less than the given threshold otherwise 0. Final weight of an edge of the overlaid graph  $G'$  is simple conjunction of the components of its vector. The major challenge of this approach is to learn a good threshold per attribute and we leave it as future work. Next, we describe three different probabilistic methods for merging components of an edge vector.

### A. Probability Mass

This technique transforms the weight  $w$  of each edge of a component graph into its probability  $p$  of occurrence in the graph. We compute a histogram of all the edge weights and then normalize it to obtain probability mass for each weight.

Probability mass captures the relative importance of an edge globally. Now, components of an edge weight vector of the overlaid graph are probabilities and can be directly compared. Missing components in a weight vector are taken to be zero. We use two functions to aggregate these probabilities. *Max* function chooses the maximum value among all components. The second function is *Avg* which computes average of all the values. Average weight of an edge of the overlaid Graph is

$$p(W, G') = \frac{1}{4} \times \sum_{1 \leq i \leq 4} (p(w, G_i))$$

We can derive direct correspondence of the *Avg* function to the marginalization of the conditional probabilities over component graphs.

$$p(W, G') = \sum_{1 \leq i \leq 4} (p(w|G_i))p(G_i)$$

If we take the probability of each component graph to be equal then the marginalized probability is just an average of all the probabilities.

### B. P-Value

P-Value is the measure of the probability of seeing a data at least as extreme as the one that is actually observed. It is used to compute the statistical significance of an observation against a null hypothesis in statistical hypothesis testing. Here, we use P-value as a measure of the weight of an edge in the graph and a high P-value means strong geo connectivity between nodes. We compute the probability mass function of edge weights of a component graph and use it to compute P-Value of each edge weight. P-value of a weight  $w_j$  is the sum of the probability masses of all  $w_i$  such that  $w_i \geq w_j$ . Components of the edge vector of the overlaid graph are now P-values. Missing components of an edge vector is taken to be zero. We use the *Max* function and the *Avg* function to aggregate the components similar to Probability mass approach. Finally, each edge of the overlaid graph gets a scalar probability value.

### C. Logistic Regression

Logistic regression is a generalized linear model used for predicting the probability of occurrence of an event having binomial distribution by fitting the data to a logistic curve. In our model, we take the components of the edge weight vector  $W$  to be explanatory variables. We take the occurrence of an edge  $e$  in the overlaid graph for a given  $W = [l, t, d, v]$  to be a Bernoulli trial. Therefore, its distribution is binomial for  $n$  trials. The probability  $p$  of the occurrence of an edge in the overlaid graph is modeled using logit as

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 l + \beta_1 t + \beta_1 d + \beta_k v$$

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 l + \beta_1 t + \beta_1 d + \beta_k v)}} \quad (1)$$

Here  $\beta = [\beta_0 \dots \beta_k]$  are unknown parameters called regression coefficients. We learn the parameters using a training set. We construct a graph whose all images are geotagged. We choose a geodesic distance threshold  $l'$ . We make a list of all the edges whose weight vector is  $W$ . The fraction of edges whose geodesic distance is less than  $l'$  in this list gives the probability of occurrence of an edge having weight  $W$ . We compute this probability for each unique weight vector in the graph to generate the training set. We use Matlab `glmfit`<sup>3</sup> function to learn the parameters from the training set. With the learned parameters, we compute the probability of existence of an edge in the overlaid graph based on its weight vector using Equation 1. Probability of an edge is computed only using the components which are present in its weight vector.

## VI. DISCOVERING GEO-CLUSTERS

Nodes in the overlaid graph, whose edges have been assigned scalar weights after integration of the components, can be clustered using graph clustering algorithms from literature. We chose a random walk based method, Markov Clustering Algorithm (MCL)<sup>4</sup> [19], for its simplicity and scalability. Random Walk is a special case of Markov chain. MCL uses the fact that a random walk of  $k$ -steps starting from a node of a dense region in a sparse graph will have high probability of finding a path of  $k$ -length which is confined in the same dense region. This property helps a random walk based algorithm to find these clusters. MCL deliberately boosts this property by an iterative procedure. It maps an input graph onto a Markov Matrix and then iteratively computes the transition probabilities via expansion and inflation. Expansion allows a node to see new neighbors whereas inflation helps to promote favored neighbors and demote other ones.

## VII. DATASET AND GRAPH CONSTRUCTION

We collected images and their meta-data from Flickr using the Flickr API service. We also collected timestamp, user Id, geotag, and text tags for each image. We scaled all the images to 400×400 pixels using the Python Imaging Library<sup>5</sup>. Our

dataset was 100,000 images. We did not perform any filtering on our dataset. Table I show that 7% of images had neither geo tag nor text tag. Next, we discuss the methodology used to construct graph for each attribute.

**Geo Graph:** We computed pairwise geodesic distance between all the 29% geotagged images in the dataset using Vincenty formula<sup>6</sup>. We assumed that the image pairs with geodesic distance of more than 100 kms are not geo-located and removed all such edges.

**Temporal Graph:** Temporal relationship is defined only between images of the same user. We grouped all the images of a user using user Id and computed pairwise timestamp difference measured in seconds. We removed all the edges where difference is greater than 15 days.

**Tag Graph:** We constructed a tag distance graph on the 91.6% of images having tags. Tags provided by users often have misspelt and non-relevant words. We performed a high level filtering to remove this noise. We prepared a set of tags given by a user across all her images. We prepared a histogram of tags using tag set of all users. We removed all those tags from all the images whose frequency was 1. We computed Jaccard's similarity coefficient between tag vectors of images. We also used Cosine and  $L_1$  distance measure. We measured precision for top 10 nearest neighbor for 10 given queries. We empirically found Jaccard's coefficient to perform better than Cosine or  $L_1$ .

**Scalable construction of Tag Graph:** The naive method to compute pairwise similarity has complexity of  $O(n^2)$  for  $n$  vectors. We made it efficient by using the fact that two vectors will have similarity greater than 0 only if they have at least one tag in common. For each tag, we prepared a list of images which were tagged with this tag. We stored these lists in a hash structure using tag as the key. For a given image tag vector, we found all the lists containing its image Id by hashing each of its tags. We computed the similarity of the given image tag vector with the tag vectors of all the unique images in the retrieved lists. Later, we removed the given image Id from all the lists. This method is a scalable methods to compute similarity graph. Finally, we took the complement of the similarity score to assign distance to each pair of images using maximum value of Jaccard's similarity which is 1. We took only top-100 neighbors having least distances for each image to address space usage. Average distance for the 100th neighbor is 0.7 in our tag graph.

**Content Graph:** We computed the visual similarity between images using SIFT [18] features and the bag of words model [20]. We scanned an image to extract affine covariant regions [21] and obtained a 128 dimensional SIFT vector per region. This feature is shown to be invariant to affine, scale, and photometric changes to limited extent and has relatively good performance for natural image matching [17].

Next, we transformed an image to a bag of words by quantization of its vectors. We randomly sampled 1 million vectors from the database and performed k-means clustering

<sup>3</sup><http://www.mathworks.com/access/helpdesk/help/toolbox/stats/glmfit.html>

<sup>4</sup><http://micans.org/mcl/>

<sup>5</sup><http://www.pythonware.com/products/pil/>

<sup>6</sup><http://www.movable-type.co.uk/scripts/latlong-vincenty.html>

Images with Geotags	28963
Images with Tags	91623
Images without Geotags and Tags	7022

TABLE I  
DATASET BREAK-UP FOR ATTRIBUTES.

with 500 randomly selected cluster centers. We repeated it 10 times and chose the cluster centers having the minimum scatter. Each cluster was assigned a label. We assigned symbols to the remaining points in the database by computing its nearest cluster center. Then, we prepared a histogram of the symbols of an image. We computed similarity between pairs of images histograms using Jaccard’s coefficient. We empirically found Jaccard’s coefficient to perform better than Cosine and  $L_1$ . We converted similarity to distance weight by taking its complement using the maximum value of Jaccard’s similarity which is 1. We used the method similar to the tag similarity computation to achieve scalability. We only retained the top 100 neighbors of an image to address space usage. Average distance for the 100th neighbor is 0.83 in our content graph.

### VIII. EXPERIMENTAL EVALUATION

Attribute Combination	No. of Images for Threshold1	No. of Images for Threshold2
Temporal	23655	23655
Tag	6896	11740
Content	175	272
Temporal + Tag	27139	29972
Temporal + Content	23798	23871
Tag + Content	7054	11966
Temporal + Tag + Content	27456	30161

TABLE II  
NUMBER OF IMAGES CONNECTED TO AT LEAST ONE GEOTAGGED IMAGE WITH SMALL DISTANCE ON DIFFERENT ATTRIBUTES. THRESHOLD1= [TEMPORAL:1 HR, TAG:0.3, CONTENT:0.3] . THRESHOLD2 = [TEMPORAL:1 DAY, TAG:0.5, CONTENT:0.5]

**Data Characterization:** We performed experiments to study the usefulness of attributes. We divided the dataset  $D$  into images having geotags ( $D_g$ ) and images having no geotags ( $D_{ng}$ ). For a combination of secondary attributes, we found all the images in  $D_{ng}$  which were linked to at least one image in  $D_g$  with an edge weight of less than the given thresholds. We performed this experiment for two set of thresholds as shown in Table II. We also measured the connectivity property of each attribute by preparing a set of images such that each image is connected to at least one other image in the set for a given threshold value. Table III gives the size of such sets. These provide evidence that combination of the attributes is effective for geo-clustering.

**Ground Truth:** To prepare a ground truth for our dataset, we manually went through the image text tags and prepared a set of 10 geographic keywords (e.g. Istanbul). For each keyword, we prepared a list of images tagged with the keyword. Thus, we obtained 10 clusters of images corresponding to 10 different geographic locations. We made an assumption that the images having the same geographic keyword belong to the same location. We created two test datasets. We named first

Attribute	Threshold	No. of Images
Geotag	50 kms	28952
Text Tag	1 day	62677
Content	0.25	21578
Temporal	0.25	87355

TABLE III  
COVERAGE OF IMAGES FOR A GIVEN THRESHOLDS.

test dataset as +Ve dataset. To form this dataset, we randomly picked one cluster out of 10 clusters. Then, we randomly picked  $k$  images from the selected cluster to generate one test point. We repeated this process to generate 1 million points. The second test dataset is called -Ve dataset. To form this dataset, we randomly picked  $k$  clusters out of 10 clusters. We picked at random one image from each of the selected clusters to generate a  $k$ -size test point. We repeated this process to generate 1 million points. We took  $k=2$  for our experiment.

**Validation Method:** We adopt a classification approach to validate our clusters. For +Ve dataset, if all the images in a test point lie in the same cluster then it is classified as true otherwise as false. For -Ve dataset, if all the images in a test point lie in different clusters then it is classified as true otherwise false. We computed the percentage (precision) of true results for both datasets. There are two worst case scenarios for algorithms that automatically decide number of clusters like MCL. They could generate only one cluster of all the images or they could put each image in different clusters. We see that precision of +Ve dataset will be 100% for the first case whereas 0% for the second case whereas -Ve dataset will have 0% precision for the former and 100% for the later. Therefore, only a high precision on both +Ve and -Ve dataset indicates a good clustering.

We apply each of the data fusion algorithms and run MCL on the overlaid graph to get clusters. Here, we study comparative results of classification of the test datasets on the resulting clusters. For every algorithm, we give results on three kind of overlaid graph (geo + temporal, geo + temporal + tag, geo + temporal + tag + content) to study the effect of progressive data fusion on cluster quality. We also present results obtained by clustering single attribute graphs after transformation of edge weights into comparable space. We observe that precision of +Ve dataset increases with progressive fusion of data irrespective of the algorithm. Thresholding results shown in Table IV give the best result of 69.40% when all the information is merged. Probability mass approach yields the best result of 80.53% with *Max* function as seen in Table V. Logistic Regression results shown in Table VII have maximum precision of 76.97% after all the data are integrated. P-Value gives maximum precision among all the algorithms which is **83.19%** for *Max* function as shown in Table VI. Precision for -Ve dataset is high for every case but interestingly decreases as we fuse more data irrespective of the algorithm. This behavior can be explained with graph density. A single attribute graph is very sparse making MCL to discover fine grained clusters. Therefore, the precision of -Ve dataset is high but it is low for +Ve dataset. New edges get added to the graph as we merge more data making the overlaid

Graph	% Precision for +Ve Dataset	% Precision for -Ve Dataset
Geo	34.15	99.91
Temporal	9.56	99.99
Tag	29.92	99.99
Content	0.08	99.99
<i>Geo + Temporal</i>	43.49	99.89
<i>Geo + Temporal + Tag</i>	69.39	99.76
<i>Geo + Temporal + Tag + Content</i>	<b>69.40</b>	99.76

TABLE IV  
CLASSIFICATION PRECISION FOR THRESHOLDING APPROACH.  
THRESHOLD=[GEO:20 KMS, TEMPORAL:1 HOUR, TAG:0.3,  
CONTENT:0.3].

Graph	% Precision for +Ve Dataset		% Precision for -Ve Dataset	
	Avg fn	Max fn	Avg fn	Max fn
Geo	34.45		99.9	
Time	22.52		99.78	
Tag	52.88		98.23	
Content	0.11		99.98	
<i>Geo + Temporal</i>	54.69	54.69	99.68	99.68
<i>Geo + Temporal + Tag</i>	78.96	80.05	96.17	90.34
<i>Geo + Temporal + Tag + Content</i>	78.95	<b>80.53</b>	92.07	91.58

TABLE V  
CLASSIFICATION PRECISION FOR PROBABILITY MASS METHOD.

Graph	% Precision for +Ve Dataset		% Precision for -Ve Dataset	
	Avg fn	Max fn	Avg fn	Max fn
Geo	35.00		99.9	
Time	22.57		99.78	
Tag	56.82		97.87	
Content	1.06		99.98	
<i>Geo + Temporal</i>	56.00	56.00	99.27	99.27
<i>Geo + Temporal + Tag</i>	79.23	81.00	96.17	90.34
<i>Geo + Temporal + Tag + Content</i>	82.16	<b>83.19</b>	93.49	89.14

TABLE VI  
CLASSIFICATION PRECISION FOR P-VALUE METHOD.

Graph	% Precision for +Ve Dataset	% Precision for -Ve Dataset
Temporal	22.61	99.71
Tag	56.93	97.90
Content	50.44	51.83
<i>Geo + Temporal</i>	54.94	98.55
<i>Geo + Temporal + Tag</i>	75.53	81.35
<i>Geo + Temporal + Tag + Content</i>	<b>76.97</b>	81.11

TABLE VII  
CLASSIFICATION PRECISION FOR LOGISTIC REGRESSION.

graph denser. It helps MCL to discover well formed clusters which increases +Ve dataset precision with a slight drop in -Ve dataset precision. Our empirical results show that we can probabilistically fuse heterogeneous data to get geo-clustered images even in case of missing geotags.

## IX. CONCLUSION AND FUTURE WORK

Geo-clustering of images is attractive from both research and commercial perspective because of its wide application. Most of the images in the community shared repositories

are not geotagged. In this paper, we proposed probabilistic techniques to fuse optional meta-data and cluster the images even in the case of missing geotags. We abstracted the problem in a general graph setting with a heterogeneous weight vector on each edge which is of general interest. Our probabilistic approaches are general and applicable to other domains. We empirically showed that better clustering is achieved after merging information from meta-data.

In the future, we would like to use better techniques like tf-idf and word stemming to select meaningful text tags. We would also like to experiment with other visual features for measuring visual similarity between images. We would also try the commonly used idea of random walk for our problem.

## REFERENCES

- [1] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3d," in *SIGGRAPH*, 2006, pp. 835–846.
- [2] J. Hays and A. A. Efros, "Im2gps: estimating geographic information from a single image," in *CVPR*, 2008.
- [3] J. Yuan, J. Luo, H. Kautz, and Y. Wu, "Mining gps traces and visual words for event classification," in *MIR*, 2008, pp. 2–9.
- [4] Kennedy, S. Lyndon, and M. Naaman, "Generating diverse and representative image search results for landmarks," in *WWW*, 2008, pp. 297–306.
- [5] D. J. Crandall, L. Backstrom, D. P. Huttenlocher, and J. M. Kleinberg, "Mapping the world's photos," in *WWW*, 2009, pp. 761–770.
- [6] T. Quack, B. Leibe, and L. Van Gool, "World-scale mining of objects and events from community photo collections," in *CVIR*, 2008, pp. 47–56.
- [7] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *ICCV*, 2009.
- [8] E. Kalogerakis, O. Vesselova, J. Hays, A. A. Efros, and A. Hertzmann, "Image sequence geolocation with human travel priors," in *ICCV*, 2009.
- [9] R. Sharan, I. Ulitsky, and R. Shamir, "Network-based prediction of protein function," *Molecular systems biology*, vol. 3, no. 88, 2007.
- [10] G. R. G. Lanckriet, T. D. Bie, N. Cristianini, M. I. Jordan, and W. S. Noble, "A statistical framework for genomic data fusion," *BMC Bioinformatics*, vol. 20, no. 16, pp. 2626–2635, 2004.
- [11] Y. Chen and D. Xu, "Genome-scale protein function prediction in yeast *saccharomyces cerevisiae* through integrating multiple sources of high-throughput data," in *PSB*, 2005.
- [12] M. Deng, T. Chen, and F. Sun, "An integrated probabilistic model for functional prediction of proteins," in *Proceedings of the seventh annual international conference on Research in computational molecular biology*, 2003.
- [13] H. Lee, Z. Tu, M. Deng, F. Sun, and T. Chen, "Diffusion kernel-based logistic regression models for protein function prediction," *A Journal of Integrative Biology*, vol. 10, no. 1, pp. 40–55, 2006.
- [14] T. Rattenbury and M. Naaman, "Methods for extracting place semantics from flickr tags," *ACM Transactions on the Web (TWEB) archive*, vol. 3, no. 1, pp. 1559–1131, 2009.
- [15] M. Cooper, J. Foote, A. Girgensohn, and L. Wilcox, "Temporal event clustering for digital photo collections," *ACM MCCA*, vol. 1, no. 3, pp. 269–288, 2005.
- [16] J. C. Platt, "Autoalbum: Clustering digital photographs using probabilistic model merging," in *IEEE Workshop on Content-Based Access of Image and Video Libraries 2000*, 2000, pp. 96–100.
- [17] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *CVPR*, 2007, pp. 1–8.
- [18] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, pp. 91–110, 2004.
- [19] S. van Dongen, "A cluster algorithm for graphs," National Research Institute for Mathematics and Computer Science in the Netherlands, Tech. Rep. INS-R0010, 2000.
- [20] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *ICCV*, 2003.
- [21] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool, "A comparison of affine region detectors," *IJCV*, vol. 65.