Deep Geo-Constrained Auto-Encoder for Non-Landmark GPS Estimation

Suhui Jiang, Yu Kong, *Member, IEEE*, and Yun Fu, *Senior Member, IEEE*

Abstract—This paper addresses the problem of geotagging images, i.e., assigning GPS coordinates (i.e., latitude, longitude) to images using image contents. Due to the huge appearance variability of visual features across the world, the images' contents and their GPS coordinates may be inconsistent. This means images captured from geographically close areas may appear visually distinct; and images with visually similar contents may be taken from geographically distant areas. In this paper, we propose a deep Geo-constrained Auto-encoder (DGAE) to solve these inconsistency problems. Using clustered GPS data and visual data, our approach identifies inconsistent data pairs (i.e., image, GPS). We then propose a novel deep learning framework that can learn similar feature representations for geographically close images and distinct feature representations for geographically distant images. We introduce two new constraints: the same-area constraint and the easy-confusing constraint to our feature learning networks. The former one penalizes images from the same area but with very distinct visual features, and the latter one penalizes images from distant areas but with very similar visual features. A deep architecture is developed to further improve learning discriminative features, which can disambiguate different geometric locations. Our approach is extensively evaluated on a newly-compiled large image geotagging dataset from large-scale community-contributed images with 664,720 images and outperforms comparison approaches.

Index Terms—GPS estimation, deep learning, auto-encoder

1 Introduction

The goal of image geotagging is to assign GPS coordinates (i.e., latitude, longitude) to a given image using its visual content. It is a very challenging task even for humans. Considering 20 example images in Fig. 1, can human easily identify where they were taken? Some of them are extremely easy. For instance, the four landmark images in the fourth row. We may easily identify that the image containing the temple was taken in Beijing. However, others are very difficult, for example, the non-landmark images in the last row. We may wonder why some of them are easy to identify but some of them are hard?

The reason why some examples are easy to identify is very likely because their visual features are distinct from others. We all know that Beijing uniquely has this type of temples. However, for difficult ones, the visual features which humans observe do not exclusively belong to a single place in the world, for example, the images in the rows of Chinatown, Building and Park in Fig. 1. They are with high visual similarity but actually belong to four different areas. In addition, there exists huge variabilities in images such as large viewpoints, scales, and appearance variations. Therefore, it is challenging for humans to geotag these images, and not to mention machines.

Image geotagging has been popularly investigated in recent years. It is due to broad applications, such as image/video retrieval [1], [2], visualization [3], [4], and tourist recommendation systems [5], [6], etc. However, most previous works only focus on restricted subsets of geo-tagging problem, such as cities with street view imageries [7], landmark buildings [8], [9], and making additional use of satellite imageries [10], [11], [12]. In contrast, our goal is to cover all kinds of locations and photos, especially non-landmark images geotagging. To our best knowledge, very few other works have addressed this task [13], [14].

Recent popular image geotagging approaches heavily rely on low-level features [13], [14], [15], [16], [17], [18], [19], [20], [21], [22]. For example, the methods in [13], [15] extract 6 types of features including edge, color, and gist features, etc. [19] extracts SIFT features and textual features for organizing a large collection of geotagged photos. An interesting work for video geotagging [23] extracts both textual features and visual features from videos. Deep learning feature representation also shows promising performance on related tasks [24].

Although satisfactory results have been shown in previous work, these approaches may fail due to the inconsistencies between images and their corresponding GPS coordinates. We observe that there mainly exist two types of inconsistencies: 1) images captured from geographically close areas may appear visually distinct, namely the same-area inconsistency; and 2) images with visually similar contents may be taken from geographically distant areas, namely the easy-confusing inconsistency. The same-area inconsistency is mainly caused by different viewpoints (e.g., aerial) of the photos of the same POI and the contexts [25] (e.g., weather) while taking photos.

- *S. Jiang and Y. Kong are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115. E-mail: {shjiang, yukong}@ece.neu.edu.*
- *Y. Fu is with the Department of Electrical and Computer Engineering, College of Engineering, and College of Computer and Information Science, Northeastern University, Boston, MA 02115. E-mail: yunfu@ece.neu.edu.*

Manuscript received 30 Nov. 2016; revised 12 Sept. 2017; accepted 17 Oct. 2017. Date of publication 20 Nov. 2017; date of current version 7 June 2019. (Corresponding author: Shuhui Jiang.)
Recommended for acceptance by Y.-S. Ong.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TBDATA.2017.2773096

Fig. 1. Non-landmark GPS estimation problem. The first three rows show that images in distant areas may appear visually similar, while images in close areas may appear visually distinct. For instance, in the third row, it is hard to identify where were the four park photos taken. Example images for landmarks and non-landmarks are shown in the last two rows. We can see how difficult our non-landmark GPS estimation problem is.

Photos of one POI are usually taken in different viewpoints such as the aerial view and the low angle view. Meanwhile, the different contexts such as daytime, night, sunny, raining, snowy may also affect the appearance of the photos. Images with variant viewpoints and contexts may be with different low-level and even mid-level features. The easy-confusing inconsistency is mainly because images in different areas may look similar, as shown in the first three rows in Fig. 1. Existing methods may easily get confused among visually similar images which are actually taken in completely different locations in the world. For example, it is hard to identify where were the four park images taken.

In this paper, we address this problem by learning more discriminative feature representation for geotagging of both landmark and non-landmark images. To learn the discriminative feature that can disambiguate different geometric locations to alleviate these inconsistencies, in our paper, a novel Deep Geo-constrained Auto-encoder is proposed. Our approach first clusters GPS coordinates of images and generates geo-clusters. Low-level visual features are also clustered into visual feature clusters. The GPS coordinate clusters and visual feature clusters demonstrate that images have similar visual contents may be geographically far away and images have different visual contents may be geographically close. The flowchart of our approach is illustrated in Fig. 2. We introduce two new constraints, the same-area constraint and the easy-confusing constraint into conventional Auto-Encoder (AE) networks. The same-area constraint penalizes images taken from the same areas but with distinct low-level visual features and the easy-confusing constraint penalizes images from distant areas but with very similar visual features. Thus, DGAE constrains images taken in close geographical areas to have similar visual features and images taken in distant areas to have distinct visual features. In the online estimation stage, given an input query image, our goal is to assign GPS coordinates to this image only using its visual content. Discriminative feature representation is encoded using offline trained DGAE network parameters. Then, visually similar images are retrieved from a large geo-tagged database for voting GPS coordinates.

The contributions of this paper can be mainly concluded as:

- We pointed out the GPS and visual inconsistency problem in a large community-contributed geo-tagged images database, which could be a bottleneck of existing non-landmark GPS estimation approaches.
- We proposed a Deep Geo-constrained Auto-encoder framework concerning the location relationship of images instead of focusing merely on vision features. DGAE adds the same-area constraint and the easy-confusing constraint on the conventional auto-encoder, which minimizes the differences of the outputs of AE within the same geo-area, and maximizes the differences of the outputs of AE in different geo-areas but with similar low-level features.

2 Related Work

Most previous works of image geo-tagging have only focused on subsets of the problem, such as cities with street view images [7], landmarks [8], [9], [26], [27], scene classification, etc. Landmark recognition systems [8], [9] recognize landmarks by retrieving matching database images and returning the landmark associated with them.

To solve the data sparsity problem in rural areas, [10], [11], [12] apply auxiliary satellite aerial images. For example, the embedding of ground images are learned from transferring knowledge from aerial images with deep networks [12].

Another task related to image geo-location is the scene recognition, for which the SUN database [28] is an established benchmark. The database consists of 131k images categorized into 908 scene categories such as “mountain”, “cathedral” or “staircase”. However, non-landmark image geo-tagging work is far more complex than scene recognition or classification.

In our paper, we target at geotagging all types of locations and photos only using visual image contents, especially for non-landmark images. To our best knowledge, very few works have addressed this task [13], [14]. Thanks for the large-scale community-contributed geo-tagged images, Hays et al. paved the way of non-landmark geotagging. Given a query photo, Im2GPS [13] and its recent extension [14] retrieve similar images from millions of geotagged Flickr photos and assign the GPS coordinates using image content of the closest match to the query. Im2GPS shows that with enough data, even this simple approach can achieve surprisingly good results.

However, existing geo-tagging works mainly depend on low-level features. For example, methods in [13], [15] extract 6 types of features such as edge, color, and gist features. Compared with existing geotagging approaches [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], our approach relies on deeply learned features, which have stronger discriminative power than low-level features such as color, SIFT, and edge, etc [29], [30].

Besides different ways of image feature representation, there are mainly two ways for GPS estimation of previous works. One is to use visual image retrieval and assign the GPS coordinates of the closest match to the query [8], [9], [13], [14]. The other way is to form the GPS estimation as a

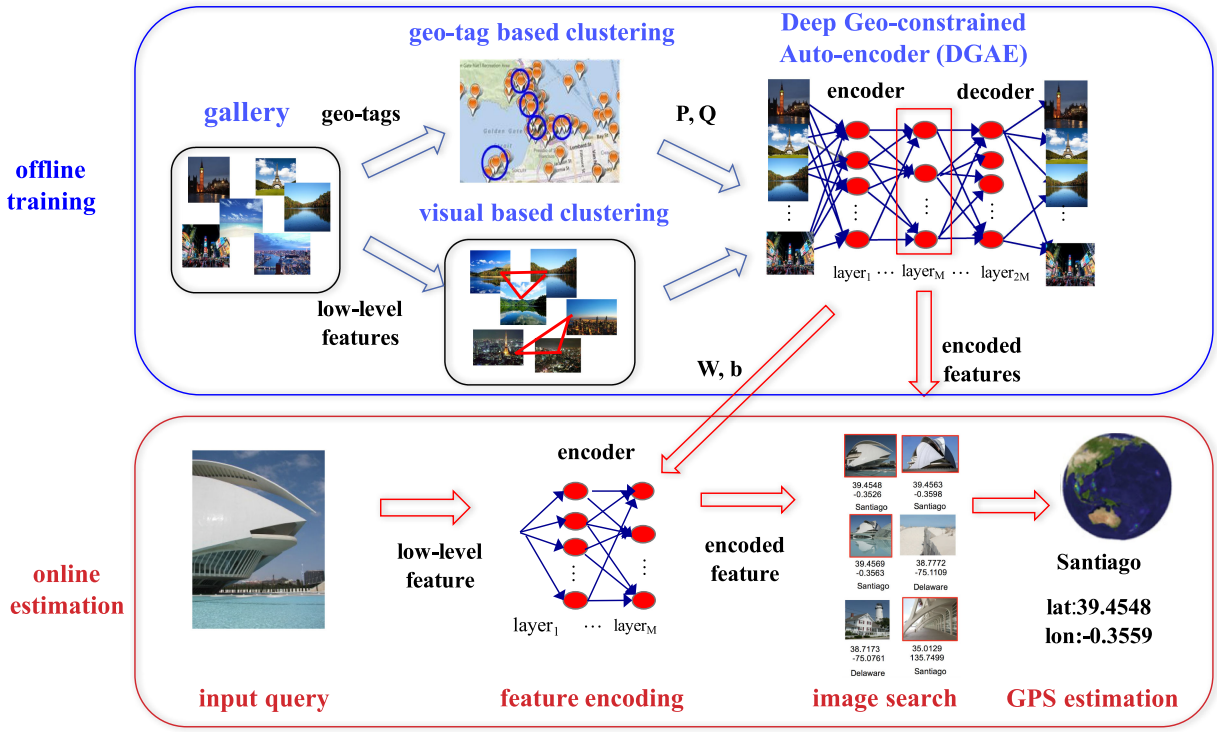


Fig. 2. System overview of the GPS estimation framework. The offline training module (blue) aims at learning a Deep Geo-constrained Auto-encoder (DGAE) network to extract high-level feature. The inputs are the world-wide image gallery including both geo-tags (i.e., latitude, longitude) and low-level visual features. First, mean-shift clustering is conducted towards geo-tags to discover geographically close areas. Second, the k nearest neighbor (k-NN) search is adopted to cluster low-level visual features to identify images with similar visual contents. P and Q are the same-area-pair index matrix and the easy-confusing-pair index matrix obtained by both geo-tags and visual based clustering results. DGAE is trained using low-level features with geo-constraints on the inconsistency pairs P and Q. The outputs of the off-line training module are encoded features and DGAE network parameters (i.e., weights W and bias b). In the online estimation module, the input is a query image only with the visual content. First, we encode the low-level feature of the query by the offline trained DGAE. Then similar images are searched based on the encoded high-level features from all the images in the gallery. Finally, GPS of this image is estimated based on the images search results. A real estimation example is provided.

classification problem [15], [24], [26], [27], [31]. [26], [27] use SVMs trained on BoVW of landmark clusters to decide which landmark is shown in a query image. Kit et al. focused on the learning a more discriminative codebook from a novel clustering method Location Aware Self-Organizing Map (LASOM) [15], [31]. LASOM learns regional similarity graph by considering both geo-tags and visual features and generates the codebook. After obtaining the codebook, given an input query image with visual feature, they search the similar codebook vectors and estimate the GPS coordinates with a weighted sum of the locations associated with similar codebook vectors.

Instead of operating on image clusters, Weyand et al. [24] predict the class of the cell of a query image's location with the visual content. The regions of these non-overlapping cells on the earth's surface are pre-defined by Google's open source S2 geometry library. However, as pointed by [24], the prediction result would be less accurate since only the cell information is provided, instead of the GPS coordinates. It is especially true for some large size cells. Thus, in order to predict accurate GPS coordinates instead of an area, we follow the visual image retrieval way [8], [9], [13], [14], which is more widely applied.

Im2GPS and its extensions [13], [14] and [10], [11], [12] are most similar to our work, which cover all kinds of locations and photos. Compared to Im2GPS [13], [14], our approach constrains (image, GPS) inconsistent data pairs, and learns visually similar features for the images taken from geographically similar areas and distinct visual features for images from geographically distant areas. This remarkably reduces the huge variability in visual features across the world, and allows us to better deal with large viewpoint, scale, and appearance variations in a large collection of images.

3 System Overview

The flowchart of our approach is illustrated in Fig. 2. In offline training, our approach takes images and their GPS coordinates (geo-tags) as inputs. Geo-tags are clustered using the mean-shift [32] algorithm to discover geographically close areas in the world. Mean-shift is most widely applied for finding highly photographed places [6], [19], [33]. Likewise, the k nearest neighbor (k-NN) search is adopted to cluster low-level visual features (we use the same features as [13], [15]) in order to identify images with similar visual content. It is expected that images with the similar content should be taken in geographically close areas. However, due to the large visual variability, we identify inconsistent (image, GPS) data pairs as the same-area-pair index matrix P and the easy-confusing pair matrix Q. Using such information, we present a Deep Geo-constrained Auto-encoder that constrains images taken in close geographical areas to have similar visual features and images taken in distant areas to have distinct visual features. We incorporate two new constraints into the proposed auto-encoder, the same-area constraint

and the easy-confusing constraint to our feature learning networks, to penalize the inconsistencies.

In the online estimation stage, given a query image, our goal is to assign GPS coordinates to this image. Low-level features are first extracted. We then compute a more discriminative feature representation using the proposed DGAE. The original low-level features and the learned hidden layer features are concatenated into a long feature vector, used as the new feature representation for the query image. A k-NN method is adopted to find the neighbors of the query image in the training data, and the GPS coordinates of the query image is computed by the GPS coordinates of its closest k neighbors.

4 Geo-Constrained Auto-Encoder

In this section, we first briefly introduce the Auto-Encoder as the preliminary of our Deep Geo-constrained Auto-Encoder. Then we describe the way to identify inconsistent image and GPS data. Third, we present DGAE in detail.

4.1 Auto-Encoder

Auto-encoder and its variants have been popularly investigated in recent years due to the effective performance in feature presentation learning [34], [35], [36], [37] and its broad applications, such as face recognition [38]. Before describing our Deep Geo-constrained Auto-encoder, we introduce the basic knowledge of AE as the preliminary [39].

Suppose that there are N images. x_i is the D dimensional feature of the ith image. Conventional AE consists of an encoder which maps the input feature to a subspace to obtain the hidden representation with a mapping function $f(\cdot)$, and a decoder which maps the hidden representation back to the original space with a mapping function $g(\cdot)$. Suppose that the dimension of the hidden representation is d. The encoding and decoding processes could be formulated as

$$z_i = f(x_i) = \sigma(W_1 \times x_i + b_1), \tag{1}$$

$$x_i = g(z_i) = \sigma(W_2 \times z_i + b_2), \tag{2}$$

where $W_1 \in \mathbb{R}^{d \times D}$ and $W_2 \in \mathbb{R}^{D \times d}$ are the linear transformation, and $b_1 \in \mathbb{R}^d$ and $b_2 \in \mathbb{R}^D$ are the biases for encoder and decoder respectively. σ is the non-linear activation function, such as sigmoid [34] in ours or $tanh(\cdot)$ function.

Conventional AE minimizes reconstruction error as following to optimize the parameters W_1, b_1, W_2 and b_2

$$\min_{\substack{W_1,b_1\\W_2,b_2}} \frac{1}{2N} \sum_{i=1}^{N} \left\| x_i - g(f(x_i)) \right\|^2 + \lambda R(W_1, W_2), \tag{3}$$

where $R(W_1, W_2) = \|W_1\|_F^2 + \|W_2\|_F^2$ is the regularizer. $\|\cdot\|_F^2$ is the Frobenius norm. λ is the weight decay parameter to suppress arbitrary large weights.

4.2 Identify Inconsistent Data

Auto-encoders have shown promising performance in learning high-level feature presentation in many fields such as object detection and images search. However, GPS estimation problem has its own challenges rather than pure vision based image search. We observe that there exists two types of inconsistencies between images and their corresponding GPS coordinates: 1) images captured from geographically close areas may appear visually distinct; and 2) images with visually similar content may be taken from geographically distant areas. Thus, without considering the original geo-location relations of images, mid/high-level vision based feature presentation may still fail to estimate accurate GPS coordinates. In this section, we introduce how to identify the inconsistent data pairs.

The inputs of this step are images and their GPS coordinates (geo-tags). For each area, first we apply mean-shift clustering towards the GPS coordinates. Mean-shift clustering is density based, which helps to find Points of Interest (POIs) [5]. POIs are the places where crowds gather and take photos. Mean-shift is widely applied for finding highly photographed places [6], [19], [33]. Mean-shift clustering does not require the assumptions of the number of clusters as most clustering methods do (e.g., K-means clustering). It only requires the assumption of the scales (i.e., bandwidth), which could be the distance between two GPS coordinates. Thus, Crandall et al. pointed out that mean-shift clustering has effective performance on GPS data which are at multiple scales [19]. However, there exists huge variabilities in images within a POI, such as large viewpoint, scale, and appearance variations. It causes the first type of inconsistency.

For a given image x_s, we calculate the distance between x_s and all the other images based on the original/low-level visual features. The top k_1 visually similar images in the same POI as x_s forms a same-area set. Each x_t in same-area set forms a same-area pair with x_s. Assuming that the same-area-pair index matrix is denoted as $P \in \mathbb{R}^{N \times N}$, and N is the number of all the images, $P_{s,t} = 1$ if x_s and x_t form the same-area pair. We set a relative large number of k_1 in order to well capture the first type of inconsistency. We would like to clarify that we do not only search for images with highly similar low-level features within the same POI to generate the same-area constraint matrix P. Our main purpose for visual based k-NN search is to find inconsistent (visual and GPS) pairs. For the first type of inconsistency (i.e., same-area inconsistency), our purpose is to not only find the most visually similar samples, but also to find some geo-closed samples but with less similarity based on low-level visual features. In this way, we force these same-area inconsistent pairs to be close to improve the image geotagging performance. If k_1 is too small, we only make samples with similar low-level visual features even closer, which is not helpful for capturing the first type of inconsistency. If k_1 is too large, we may force two exactly different images to be the same. It may degrade the reconstruction learning part of AE. In the experiments, we discuss the performance across different settings of k_1 and select k_1 according to the cross-validation.

Images visually similar but within different POIs are regarded as the second type inconsistency. For a given image x_s, the top k_2 visually similar images in different POIs from x_s form a easy-confusing set. Each x_t in easy-confusing set forms the easy-confusing pair with x_s. Assuming that easy-confusing pair matrix $Q \in \mathbb{R}^{N \times N}$, and N is the number of all the images, $Q_{s,t} = 1$ if x_s and x_t belong to easy-confusing pairs.

4.2.1 Computational Cost and Fast Solution of Generating Graph Constraints

Here we first discuss the computational cost of generating P and Q and then propose a fast solution for generating Q.

After obtaining the POIs by GPS coordinates based clustering, for generating P and Q, we build a visually similarity graph using low-level visual features. Calculating all pairwise similarities is the most naive method for generating the k-NN graph, which is with $O(n^2)$ time complexity. It means that the computational cost is directly related to the number of samples in the similarity graph. For calculating P, we only need to build the graph within each POI, which is with limited number of samples. As a result, the computational cost of building the complete k-NN graph to calculate P is small. However, for calculating Q, we need to build the complete k-NN graph with all the samples outside the POI. To the large-scale dataset, the computational cost could be very high.

In this section, we provide a fast solution for identifying the second type of inconsistent data pairs and generating Q. The way of identifying the first type inconsistent data pairs keeps the same. It is enlightened by Anchor Graph Regularization based methods [40], [41], [42]. As shown in previous works, Anchor Graph Regularization achieves more efficient computational cost than the complete k-NN graph, while persevering the competitive performance. The main idea of the fast solution is straightforward. Instead of conducting the k-NN search completely pairwise, we generate the graph in which the nodes are only anchor points, which are the centroids of visual feature based clustering.

Within each POI, we conduct mean-shift clustering based on low-level visual feature similarity using euclidean distance. Here we discuss the bandwidth of mean-shift clustering from 0.001 to 0.1 empirically. We set the bandwidth at 0.01 by manually checking the performance of randomly picked 50 visual clusters. For each POI, there are usually 1 to 5 visual clusters. After obtaining the visual based clusters, we regard the centroids of clusters as anchors and build the pairwise graph where only these anchor points are regarded as graph nodes. By less nodes in the graph, we obtain a lower computational cost at $O(n^{*2})$, which n^* is the number of anchors and $n^* << n$.

For generating Q, for a given image x_i in one POI, we first find the top k_3 anchors outside this POI but with highly similar visual features. Then we calculate the distance between x_i and the samples in these k_3 clusters. We pick the top k_2 samples to generate Q, and we use the same number of k_2 as the original graph generating method. Empirically, when $k_3 = 50$, the samples in these k_3 clusters could already cover 90 percent of the top k_2 images with the completely k-NN search. Thus, Q generated by the fast method changes slightly and we could achieve almost the same performance with far lower computational cost. For example, when $k_2 = 20$, we only need to sort about 1,000 images.

4.3 Deep Geo-Constrained Auto-Encoder (DGAE)

In this section, we introduce our Deep Geo-constrained Auto-Encoder. Fig. 3 provides the intuitive illustration of our scenario and how DGAE works. The inputs are nine samples from x_1 to x_9, and each three samples (e.g., x_1, x_2, x_3) are in one geo-closed area. Each sample is presented with a graph with different color and shape. Assuming that the shape illustrates the low-level feature and the color illustrates mid/high-level feature, as shown in the right part of Fig. 3, two images of the same area may be with different low-level features. For example, the last two graphs are all in Santiago but with triangular and square shapes. Meanwhile, two middle images are with similar low-level feature but in two different cities.

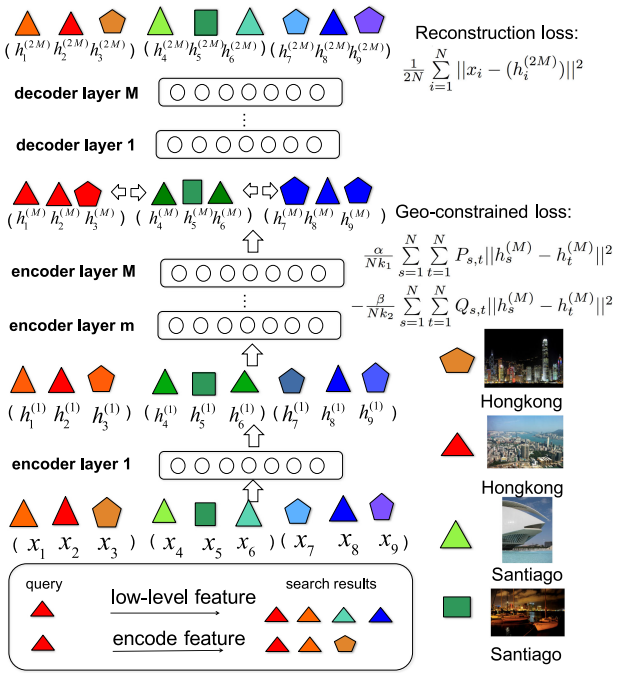


Fig. 3. Intuitive illustration of DGAE (color version preferred). Each graph presents an image. Nine graphs from x_1 to x_9 in three groups (within a "( )") are shown as example inputs. Each group presents one geo-area. The shape of the graph (e.g., triangle, square and pentagon) presents low-level feature and the color (e.g., red, green and blue) presents encoded mid-/high-level feature. Through training processing, colors in the same group become more similar, and the color difference between groups become more remarkable (three-primary colors). It shows that the encoding process enhances the discrimination of features in different areas.

It is not difficult to think to use deep learning networks, such as Auto-Encoder [34] to learn mid/high-level feature representations. Although the mid/high-level features by conventional deep networks achieve high performance than low-level features, we still face the problem that, images in geographically close area may be with visual distinction and images with visually similar content may be taken from geographically distant areas. In Fig. 3, we could see that the colors of three samples in one area may be distinct, and the colors of x_6 and x_7 may be similar.

Graph embedding frameworks and the extensions [40], [41], [42], [43] enforce the intra-class compactness and the inter-class separability. It is recently introduced in deep metric learning [44] in face recognition. However, few works have enforced graph embedding in auto-encoder, not to mention geo-constrained auto-encoder for non-landmark geo-tagging problem.

Inspired by graph embedding, our geo-constrained auto-encoder takes advantage of two geo-constraints: the same-area constraint and the easy-confusing constraint to penalize the inconsistencies between images and GPS. The same-area

constraint minimizes the differences of the outputs of AE within the same geo-area, in order to constrain images in close geographical areas to have similar mid/high-level features. Easy-confusing constraint maximizes the differences of the outputs of AE in different geo-area but with similar low-level feature, to constrain images taken in distant areas to have distinct visual features.

The inputs of DGAE are low-level features of images in the gallery, together with the same-area-pair index matrix P, and the easy-confusing-pair index matrix Q. We have already described the way to calculate P and Q in the last section. The outputs are the learned deep network with weights W and biases b. Assuming that there are M encoding layers and M decoding layers, we unify the notations W_1 and W_2 in Eqs. (1) and (2) as $W^{(m)}$ and b_1 and b_2 as $b^{(m)}$ of the mth layer, $(m = 1, 2, \ldots, 2M)$. Compared with conventional AE, we add geo-constraints on top of encoder layers (Mth layer) and formulate a supervised auto-encoder method as

$$\begin{aligned}\min_{W,b} J &= J_1 + \alpha J_2 + \beta J_3 + \gamma J_4 \\ &= \frac{1}{2N}\sum_{i=1}^{N} ||x_i - g^{(M)}(f^{(M)}(x_i))||^2 \\ &+ \frac{\alpha}{Nk_1}\sum_{s=1}^{N}\sum_{t=1}^{N} P_{s,t}||f^{(M)}(x_s) - f^{(M)}(x_t)||^2 \\ &- \frac{\beta}{Nk_2}\sum_{s=1}^{N}\sum_{t=1}^{N} Q_{s,t}||f^{(M)}(x_s) - f^{(M)}(x_t)||^2 \\ &+ \gamma R(W, b),\end{aligned} \tag{4}$$

where J_1 minimizes the reconstruction error of all the samples. J_1 and J_4 are the same as conventional auto-encoder.

Different from conventional auto-encoder, we add a new constraint J_2 working as the same-area constraint and J_3 working as the easy-confusing constraint. Two parameters α ($\alpha > 0$) and β ($\beta > 0$) balance the importance between reconstruction error J_1, the same-area constraint J_2 and the easy-confusing constraint J_3. P and Q are indicators for the same-area pair and the easy confusing part, which are described in the last section.

The output of x in the mth layer ($m = 1, 2, \ldots, 2M$) could also be described as

$$f^{(m)}(x) = h^{(m)} \tag{5}$$

$$= \sigma(W^{(m)}h^{(m-1)} + b^{(m)}). \tag{6}$$

In Fig. 3, through training, the same-area constraint J_2 minimizes the difference of mid-/high-level features. The easy-confusing constraint J_3 maximizes the difference of samples' colors with similar shape but in a different area. Thus, the colors in the same group progressively change to be the same, and meanwhile are with maximum difference with the color in different areas. For example, orange changes to red and purple changes to blue. For a query graph, for example, a red triangle, if we only use the low-level feature (shape) to search for similar graphs, the results may be with the same shape but in different colors/geo-areas. However, with the encoded feature, the shape attribute is ignored and results are with similar colors and within the same geo-area.

4.4 Solutions

In this section, we provide the solution for solving the objective function Eq. (4). Stochastic gradient descent + back propagation [45] is applied for optimization. The basic rules for updating $W^{(m)}$, $b^{(m)}$ $(m = 1, 2, \ldots, 2M)$ of the mth layer of auto-encoder with M encoded layers and M decoded layers will be

$$W^{(m)} := W^{(m)} - \frac{\partial}{\partial W^{(m)}} J(W, b), \tag{7}$$

$$b^{(m)} := b^{(m)} - \frac{\partial}{\partial b^{(m)}} J(W, b). \tag{8}$$

The gradients of the objective function J in Eq. (4) with respect to the parameters $W^{(m)}$ is computed as

$$\begin{aligned}\frac{\partial J}{\partial W^{(m)}} &= \frac{2}{N}\sum_{i=1}^{N} L_i^{(m)} h_i^{(m-1)^T} \\ &+ \frac{2\alpha}{Nk_1}\sum_{s=1}^{N}\sum_{t=1}^{N} P_{s,k}\big(L_{s,t}^{(m)} h_s^{(m-1)^T} + L_{t,s}^{(m)} h_t^{(m-1)^T}\big) \\ &- \frac{2\beta}{Nk_2}\sum_{s=1}^{N}\sum_{t=1}^{N} Q_{s,k}\big(L_{s,t}^{(m)} h_s^{(m-1)^T} + L_{t,s}^{(m)} h_t^{(m-1)^T}\big) \\ &+ 2\gamma W^{(m)}.\end{aligned} \tag{9}$$

And the gradients of $b^{(m)}$ is computed as

$$\begin{aligned}\frac{\partial J}{\partial b^{(m)}} &= \frac{2}{N}\sum_{i=1}^{N} L_i^{(m)} + \frac{2\alpha}{Nk_1}\sum_{s=1}^{N}\sum_{t=1}^{N} P_{s,k}(L_{s,t}^{(m)} + L_{t,s}^{(m)}) \\ &- \frac{2\beta}{Nk_2}\sum_{s=1}^{N}\sum_{t=1}^{N} Q_{s,k}(L_{s,t}^{(m)} + L_{t,s}^{(m)}) + 2\gamma b^{(m)},\end{aligned} \tag{10}$$

where $L_i^{(m)}$, $L_{s,t}^{(m)}$ and $L_{t,s}^{(m)}$ are computed as follows:

$$\begin{aligned}L_i^{(2M)} &= (x_i - h_i^{(2M)}) \odot \sigma'(z_i^{(2M)}), \\ L_{s,t}^{(M)} &= (h_s^{(M)} - h_t^{(M)}) \odot \sigma'(z_s^{(M)}), \\ L_{t,s}^{(M)} &= (h_t^{(M)} - h_s^{(M)}) \odot \sigma'(z_t^{(M)}), \\ L_i^{(m)} &= (W^{(m+1)^T} L_i^{(m+1)}) \odot \sigma'(z_i^{(m)}), \\ L_{s,t}^{(m)} &= (W^{(m+1)^T} L_{s,t}^{(m+1)}) \odot \sigma'(z_s^{(m)}), \\ L_{t,s}^{(m)} &= (W^{(m+1)^T} L_{t,s}^{(m+1)}) \odot \sigma'(z_t^{(m)}).\end{aligned}$$

The operation $\odot$ denotes the element-wise multiplication, and $z_i^{(m)}$ is given as $z_i^{(m)} = W^{(m)}h_i^{(m-1)} + b^{(m)}$. Note that during back-propagation, only the reconstruction loss has effects on the optimization of decoder layers, and both the reconstruction loss and geo-constrained losses have effects on the encoder layers.

Algorithm 1 illustrates the solution for learning the DGAE. After optimizing the DGAE, we use the output of the top hidden layer $h^{(M)}$ as the learned high-level representation of the image.

4.5 Geometric Interpretation of DGAE

In this section, we interpret DGAE from the geometric perspective under the manifold assumption [46] inspired by

the manifold learning perspective regarding to the insight behind the Denoising Auto-encoder (DAE) [34].

Algorithm 1. DGAE

INPUT: Low level features of all the training images x; Intra class index matrix P; Inter class index matrix Q; Parameters: α, β, γ, M, learning rate λ, convergence error ε, and total iterative number T.

OUTPUT: Weights and biases $\{W^{(m)}, b^{(m)}\}_{m=1}^{2M}$.

for $k = 1, 2, \ldots, T$ **do**
  Do forward propagation to all the training data and get $h^{(m)}, m \in \{1, \ldots, 2M\}$ for each layer.
  Compute the training loss J in the kth iteration using Eq. (4) and denote as J_k.
  for $m = 2M, 2M-1 \ldots, 1$ **do**
    Calculate $\partial J_k / \partial W^{(m)}$ and $\partial J_k / \partial b^{(m)}$ by back-propagation using Eqs. (9) and (10).
  end for
  for $m = 1, 2, \ldots 2M$ **do**
    $W^{(m)} := W^{(m)} - \lambda \partial J_k / \partial W^{(m)}$;
    $b^{(m)} := b^{(m)} - \lambda \partial J_k / \partial b^{(m)}$;
  end for
  $\lambda \leftarrow 0.95 \times \lambda$
  if $|J_k - J_{k-1}| < \varepsilon$, go to Output
end for

In our visual based GPS estimation task, we suppose that the images X with visually similarity of one POI lie close to a low dimensional manifold. Images $\tilde{X}_1$ with dissimilar low-level features but within one POI are likely being far from the manifold. Images $\tilde{X}_2$ with similar low-level features but in different POIs are likely being close to the manifold. If we only apply low-level features in the original space, the distance between $\tilde{X}_1$ and X is likely to be larger than the distance between $\tilde{X}_2$ and X.

In DGAE, on one hand, in the first regularization term, a stochastic mapping $q_1(\tilde{X}_1|X)$ manages to find $\tilde{x_1} \in \tilde{X}_1$ in X to obtain the corrupted the vision of x as $\tilde{x}_1$. During the training processing, similar to DAE, DGAE learns the stochastic operator $p_1(X|\tilde{X}_1)$ while generating P that maps the images whose low-level features are dissimilar to the images close to the manifold back to the manifold.

On the other hand, in the second regularization term, $q_2(\tilde{X}_2|X)$ manages to find $\tilde{x}_2 \in \tilde{X}_2$ in X to obtain the corrupted vision of x as $\tilde{x}_2$. During the training processing, DGAE learns the stochastic operator $p_2(X|\tilde{X}_2)$ while generating Q that maps the images whose low-level feature are similar to the images but of different POIs far away from the manifold.

4.6 Discussion of Arriving New Images

Users continuously upload images to social media such as Flickr.com every day, and the centers and slopes of cityscapes and landscapes change gradually. There would be some new POIs appearing and some old POIs disappearing and the centers and scopes of the POIs may also be changing gradually. We assume that images are uploaded within one time period (i.e., one month) would not change a lot. Thus, we do not need to update the centers of POIs and the visual based anchors within one time period. After one period, based on the data, we completely re-mine the POIs, re-generate the graph and re-train the model.

During each time period, we fine-tune the model with the continuously arriving new images each day. Here we introduce the way of fine-tuning the deep learning model.

After a new image arriving, we first assign this image to one POI based on the GPS coordinate. Then, we update the P and Q matrix by calculating the columns and rows containing this new image in either the original or fast graph generating ways. After updating P and Q, we apply the stochastic gradient decent (SGD) and back propagation for updating the parameters of each layer. Thus, we do not need to calculate the gradient decent of all the samples and then update the weights. Instead, with SGD, we only update the weights for the new coming images.

5 GPS Estimation with DGAE

In the online-estimation stage, given an input query image, our goal is to assign GPS coordinates to this image only based on its visual feature. What we want to emphasize is: we provide the GPS coordinates (i.e., latitude, longitude) estimation and it is more accurate than using classification methods to classify a query image to an area (i.e., city) [15], [24].

Following [13], [14], we estimate the GPS for the query image through retrieving visually similar images and assign the GPS of best matches to the query image.

As shown in Fig. 2, it mainly consists of three steps. First, low-level features are extracted from the query image and concatenated as a long feature vector denoted as x. We encode low-level feature x with DGAE to obtain mid/high level feature. Assuming that a DGAE is with M encoding layers, for the mth hidden layer ($m = 1, 2, \ldots, M$), x would be encoded through $f^{(m)}(x) = h^{(m)} = \sigma(W^{(m)} h^{(m-1)} + b^{(m)})$. We stack all the hidden layers together to present the query image. Thus, the query image and geo-tagged image set are mapped into the same feature space.

Second, a k-NN based search is adopted to find the neighbors of the query image from the big geo-tagged community-contributed dataset with encoded mid/high-level features. The key for accurate image retrieval is the robust feature representation.

Third, two heuristics of models are conducted after k-NN search: *1-NN* and *Mean-Shift*, following [13]. 1-NN model applies the GPS coordinate of top nearest neighbor as the GPS estimation result. We apply Euclidean distance similarity to measure the distance between two images. For Mean-Shift model, after k-NN search, a mean-shift clustering with bandwidth of 500 km is conducted towards top k ($k = 120$) nearest neighbors. We capitalize each word in "Mean-Shift" heuristic to make it distinguishable from "mean-shift" clustering. The centroid of the cluster which owns the highest cardinality is applied as the GPS estimation result.

6 Experiments

In this section, we conduct experiments comparing recent state-of-the-art works of non-landmark GPS estimation under a newly collected large database. The estimation performance of our methods and the state-of-art methods are shown, followed by the discussions of parameters, visualization and computational cost.

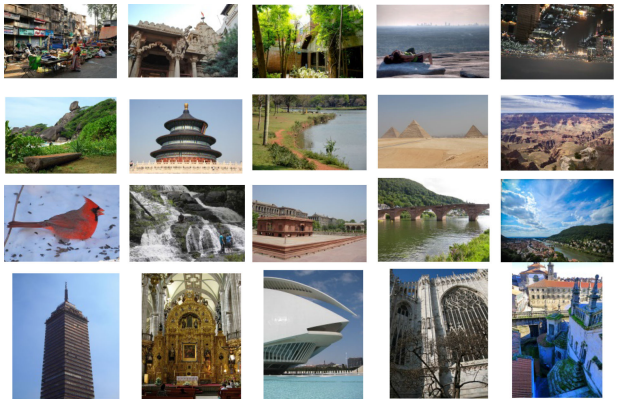


Fig. 4. Examples of our test dataset of non-landmark images for GPS estimation.

6.1 Dataset and Data Processing

We collect a new less biased dataset for *non-landmark* GPS estimation task from geo-tagged community contributed photos. Although there are a few GPS estimation datasets, these datasets mainly only focus on landmark estimation [47], scene recognition [28] or cover limited cites [15], instead of focusing on world-wide non-landmark image geo-tagging. Im2GPS dataset [13] has biases towards the city centers, while our dataset covers most of the areas in/around the city, and it is less biased.

We download geo-tagged images in/around 35 areas with Flickr open API. Note that, we do not only focus on the landmarks when collecting data. All the GPS coordinates are treated equally to achieve the unbiased characteristic of our dataset. First, we remove user profiles by face detection. There are 664,720 images remained after filtering. The number of remaining images of each city is shown in Table 1.

We extract six low level-features for each image following Im2GPS [13] and LASOM [15]: two tiny version images with sizes $5 \times 5 \times 3$ and $16 \times 16 \times 3$; a CIE L*a*b color histogram with $14 \times 14 \times 4$ bin; a texture histogram with 512-bin; an edge length and edge angle histogram with 232-bin; a 'gist' vector with 600 dimension. For each image, we concatenate the six features together as a 2,893 dimension vector.

For the test set, we first randomly select 700 images from the whole dataset. There is no overlap between the training and test set. Then we filter out unusual images (e.g., black and white or artistic) to create a test set of 300 images. The number of test set is similar to [13], [14]. Fig. 4 shows 15 examples of the test dataset, which demonstrates the difficulty of estimating the GPS of most of the images.

This dataset would be publicly available upon the acceptance of this paper. Our release would contain images after filtering, meta-data of the each image and low-level features, so that researchers could save time of data preprocessing and focus on the model development. Furthermore, this dataset could not only be used for im2gps problem, but also be used for POI recommendation, POI sequence planning and so on. Here we summarize the main challenges of this dataset:

First, it is a large-scale dataset with 35 areas. It contains 664,720 images after removing the images containing faces. The dataset consists of both landmark and non-landmark images. Second, community-contributed photos are more challenging since they are with more noises, low-quality, and large variation. Third, the dataset is less biased to city centers and landmark images. Meanwhile, as we have pointed out, there are data inconsistency problems between visual and GPS coordinates.

TABLE 1
Number of Images in Each City

city	number	city	number
Ahmedabad	1,230	Houston	20,833
Arkansas	23,228	Istanbul	46,281
Atlanta	15,180	Jakarta	7,396
Bangkok	5,910	LosAngeles	14,539
Beijing	15,470	Melbourne	9,401
Bogota	13,274	MexicoCity	7,318
Cairo	13,653	NewYorkCity	27,369
Chengdu	6,300	Miami	2,301
Chicago	8,966	Milan	9,614
Colorado	14,229	Moscow	19,482
Connecticut	28,516	Osaka	12,316
Delaware	19,692	Rome	10,342
Delhi	13,459	Salvador	16,161
Florence	21,507	Santiago	44,920
Galapagos	21,123	SaoPaulo	18,894
Hawaii	21,742	Shanghai	11,759
Heidelberg	6,398	Tokyo	94,003
HongKong	41,914	-	-

6.2 Compared Methods

Although GPS estimation is a hot topic in recent years, as discussed in the related work section, few works [13], [14] have focused on the same scenario as ours: estimating the GPS coordinates of a single query image worldwide, with only visual features as input. For example, [10], [11] focus on ground-to-aerial geo-localization, which is localizing a ground-level query image by matching it to a reference database of aerial imageries.

We compared our method with two state-of-the-art works and named as LF [13] and NFLL [14]. To evaluate the effectiveness of DGAE, we also implement two deep learning frameworks with the conventional Auto-Encoder [34] and deep neural networks (CNN) [30] as two baseline methods for DGAE. Since our paper mainly focused on feature representation learning, we apply the same way of GPS estimation (k-NN based search) for LF and the deep learning comparison methods, but with different feature representations. The descriptions of comparison methods are shown as follows:

- *Low-level feature based method (LF)* [13]. Im2PGS is the first work dealing with non-landmark GPS estimation. A purely data-driven scene matching approach is proposed based on 6 low-level features.
- *New feature + Lazy Learning (NFLL)* [14]. NFLL is the extension of LF [13]. NFLL also applies low-level feature representations to search k nearest neighbors first. Then NFLL adds SIFT descriptors to enhance local feature representation based on SIFT points match between image pairs. Furthermore, a lazy learning method is proposed based on SIFT points match between image pairs.
- *Auto-Encoder based method* [34]. In order to evaluate the effectiveness of DGAE, we also investigate a

TABLE 2
Percentage of Test Images Whose Estimation Error Is within the City, Region, Country and Continent Level by 1-NN Model

Performance (%)	city	region	country	continent
LF[13]	8.7	15.3	22.9	25.3
AE	10.3	23.6	26.3	35.2
CNN	12.9	23.3	28.9	38.8
DGAE	**14.2**	**25.2**	**30.5**	**41.8**
fDGAE	14.0	**25.2**	29.9	40.7

The best and the second best results are shown with bold font style and underline.

conventional Auto-Encoder based framework. We substitute DGAE with AE which has no geo-constraints. In off-line training module, only low-level features are needed as the inputs of auto-encoder and we need not conduct mean-shift clustering towards the geo-tags.

- *Deep Convolutional Neural Networks based method* [30]. We investigate a Convolutional Neural Networks based framework. We extract deep features for both training and testing images, using AlexNet [30], which is pre-trained on ImageNet dataset. We use the outputs of fc7 layer after activation as the feature representation.
- *Deep Geo-constrained Auto-Encoder based method*. In this method, when identifying the inconsistent data pairs, we apply the original (non-fast) version described in Section 4.2. Then we apply DGAE to learn the feature representation.
- *Fast Deep Geo-constrained Auto-Encoder based method (fDGAE)*. Compared with DGAE, the fast solution of identifying the inconsistent data pairs in Section 4.2.1 is applied. Other steps are the same as DGAE.

Although we use a new dataset, the experiments are fair. Because, we implement existing works under the same settings and compare them under the same metrics on our new dataset as [14]. We compare the 6 methods under two heuristics of models of GPS estimation as described in "GPS Estimation with DGAE" section: 1-NN and Mean-Shift models following [13]. For Mean-Shift model, k nearest neighbors (e.g., k=120) form an implicit estimation of geographic location. We project the geo-locations as 3D points on the earth's surface when calculating the distance during mean-shift clustering. We use a mean-shift clustering algorithm with bandwidth of 500km. Clusters with fewer than 4 matches are disregarded. The centroid of the cluster with the highest cardinality is reported as the estimated GPS coordinate.

6.3 Ground Truth and Criteria

To evaluate the performance of GPS estimation by different methods, we calculate the distance between the estimated GPS and the ground truth GPS with a spherical law of cosines.[1] The spherical law of cosines is used generally for computing great-circle distance between two pairs of coordinates on a sphere:

$$Err = R * \arccos(\sin(lat^{(g)})\sin(lat^{(e)}) + \cos(lat^{(g)})\cos(lat^{(e)}) * \cos(lon^{(g)} - lon^{(e)})), \quad (11)$$

1. http://www.movable-type.co.uk/scripts/latlong.html

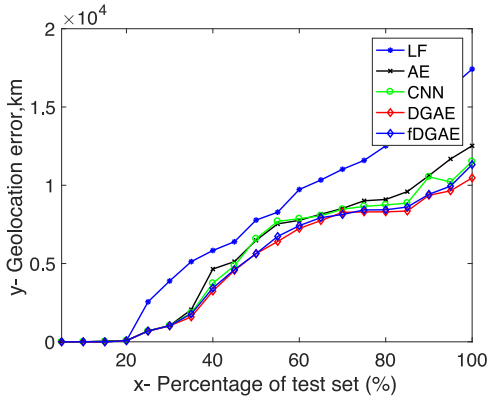


Fig. 5. Geo-location error Err with 1-NN model under LF, AE, CNN, DGAE and fDGAE. Following [13], errors are sorted from best to worst independently for each curve, thus showing the proportion of images geo-located within an error threshold. NFLL only applies Mean-Shift model. Under 1-NN model, NFLL degrades to LF. We also conducted a chance-random method, which is a baseline in [13]. Similar to [13], chance-random performs far less accurately than LF. The errors of less than 1 percent images are within 5 km and only 20 percent are within 5,000 km. Thus, we do not regard chance-random as a main comparison and show it on the figure.

where $lat^{(g)}$ and $lon^{(g)}$ are the latitude and longitude of ground truth GPS coordinates. $lat^{(e)}$ and $lon^{(e)}$ are the latitude and longitude of estimated GPS coordinates.

Ground truth GPS coordinates are either manually labeled by users when uploading images to Flickr, or automatically recoded by digital devices. R= 6371.004 km is the average radius of the earth. Err is the error of the distance between the estimated GPS and the ground truth GPS. According to [13], $Err \leq$25 km presents the error distance which is within a "City" level; $25 < Err \leq 200$ km presents the "Region" level; $200 < Err \leq 750$ km presents the "Country" level and $750 < Err \leq 2500$ km presents the "Continent" level.

6.4 Experimental Results

Figs. 5 and 6 show the error Err of geo-location estimates (as Eq. (11)) across the test set with 1-NN and Mean-Shift model by LF, NFLL, AE, CNN, DGAE and fDGAE. Also, following [13], in Table 2 and 3, we present the percentage of test images whose estimation errors are within city, region, country and continent level. We set $\alpha = 0.2$, $\beta = 0.1$, $\gamma = 0.05$, $k_1 = 20$, $k_2 = 10$, bandwidth of Mean-Shift clustering as 500 km according to cross-validation results. We set both α and β from 0 to 1 with interval 0.05.

From the observation of both Figs. 5, 6 and Tables 2, 3, we can reach the following conclusions:

1) In both 1-NN and Mean-Shift heuristics, the proposed DGAE and fDGAE achieve the best and the second best performance. The overall error (the area under the curve) of comparison methods was shown in Figs. 5 and 6. Clearly, the area on the right hand side contributes most. Our method significantly reduces the overall geo-location error, especially when x-axis ranges from 25 to 100 percent. The area of DGAE is 1.7 and 1.5 times larger than the area of state-of-the-arts in 1-NN and Mean-Shift models respectively.
2) Under all the conditions, the deep learning methods (AE, CNN, DGAE and fDGAE) perform better than low-level feature based methods (LF and NFLL).

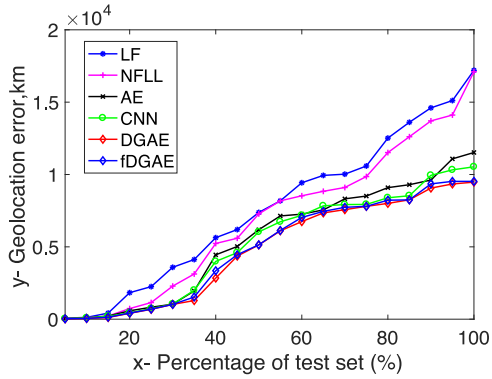


Fig. 6. Geo-location error Err with Mean-Shift model under LF, NFLL, AE, CNN, DGAE and fDGAE. Other settings are the same as Fig. 5.

3) In all cases, DGAE and fDGAE perform better than AE and CNN, which means the same-area constraint and easy-confusing constraint proposed in this paper are effective.
4) Comparing fDGAE with DGAE, fDGAE achieves very competitive or slightly lower performance than DGAE. It demonstrates that fDGAE would achieve effective accuracy with much lower computational cost, which makes it a candidate tool for large-scale dataset.
5) Similar to the observation in Im2gps [13], 1-NN approach performs better for precise localization (e.g., within a city), while Mean-Shift model performs better for global localization. We could see that generally, 1-NN approach performs better at city and region level, and Mean-Shift model performs better at country and continent levels.

6.5 Discussion of Parameters

In this section, we discuss the parameters used in POI mining and DGAE. We describe how we set these parameters and analyze the performance of different settings.

Note that the parameter discussions are conducted under the original DGAE (non-fast version). In this way, it could focus on demonstrating the influence of the parameter itself, instead of being influenced by both the parameter and the fast variation simultaneously. Although a fast version DGAE is proposed, the computational cost of the original DGAE is still affordable which is shown in Section 6.7 "Discussion of Computational Cost". Thus, it is more encouraging to use DGAE when computational cost is not a main concern, while the fast version is encouraged to use when the computing resource is limited.

TABLE 3
Percentage of Test Images Whose Estimation Error Is within the City, Region, Country and Continent Level by the Mean-Shift Model

Performance (%)	city	region	country	continent
LF[13]	3.7	12.4	17.3	26.8
NFLL[14]	6.3	15.2	20.9	32.4
AE	7.7	16.2	24.6	36.8
CNN	8.9	19.3	26.2	39.4
DGAE	**11.6**	**23.4**	32.9	**45.4**
fDGAE	10.1	22.3	**33.1**	44.6

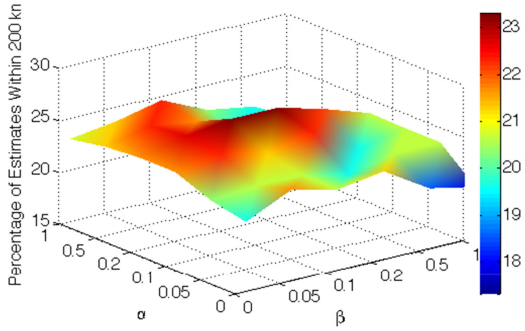


Fig. 7. Geo-localization performance across different settings of parameters α and β of DGAE.

6.5.1 Discussion of POI Mining

The only parameter we set in POI mining is the bandwidth of mean-shift clustering. It is set empirically according to related works [6], [33] and our observations of dataset. We manually check the visual images in the clusters and adjust the bandwidth. We get the same observations of the bandwidth setting as the existing works. With a very large bandwidth, one cluster may contain multiple POIs, while with a very small bandwidth, one POI may be divided into multiple clusters. We observe that the clustering results are not very sensitive to the bandwidth settings, and we could obtain satisfactory clustering performance when setting the bandwidth in the range of 0.0001 to 0.05. In our experiments, we set the bandwidth at 0.005 for POI mining.

6.5.2 Discussion of α, β and γ

In Fig. 7, we discuss the impacts of parameters α and β. We fix γ=0.05, k_1=20 and k_2=10. The percentage of correct estimates within 200 km ranges from 17.333 to 23.333 percent. When $\alpha = 0.2$, $\beta = 0.1$ and $\alpha = 0.2$, $\beta = 0.2$, DGAE achieves the highest performance at 23.333 percent.

When α=0 and β=0, since the geo-constraints have no effects on training the deep network, DGAE degrades to the general auto-encoder, and achieves 19.667 percent of correct estimation within 200 km. The highest performance of DGAE 23.333 is 3.665 percent higher than this setting and the average performance is about 2 percent higher than this setting. It indicates the effectiveness of the geo-constraints. When $\alpha \neq 0$ and β=0, only the same-area constraint has effects on network training. DGAE achieves 20.333, 21.667, 21.667, 21.667 and 20.6667 percent when α=0.05, 0.1, 0.2, 0.5 and 1. It indicates that the same-area constraint has the positive effects on the performance. When $\beta \neq 0$ and α=0, only the same-area constraint has effects on network training. DGAE achieves 21.337, 20.667, 21.333, 18.667 and 17.6667 percent when β=0.05, 0.1, 0.2, 0.5 and 1. It indicates that the easy-confusing constraint achieves positive effects when allocating a small weight. When the weight of easy-confusing constraint is too large, the performance degrades, since it may affect the learning of the reconstruction error term J_1.

The weight decay parameter γ is set empirically according to previous studies on [48], [49]. Both conventional AE and DGAE are not sensitive to γ. The performance is satisfactory when γ is in the range from 0.01 to 0.1. We pick 0.05 in our model.

TABLE 4
Discussion of k_1 and k_2 in DGAE

	0	5	10	15	20	25	30
k_1	20.2	22.3	24.2	24.1	25.2	24.3	22.1
k_2	23.7	24.8	25.2	24.8	24.5	24.5	23.8

We show the performance of the percentage of correct estimations within 200 km.

6.5.3 Discussion of Layer Size

Fig. 8 discusses different settings of layer sizes of DGAE. 0 in x-axis is the baseline, meaning that only low-level features are used. The settings of layer size are 1) [100]; 2) [2000, 1000]; 3) [2000,1000,500]; 4) [2000,1500,1000,500] and 5) [2000, 1600, 1200, 800, 500]. We see that DGAE achieves the highest performance when the number of hidden layers is 3.

6.5.4 Discussion of k_1 and k_2

In this section, we discuss the impacts of k_1 and k_2 in DGAE. In Table 4, we show the performance of the percentage of correct estimations within 200 km when k_1 and k_2 in the range from 0 to 30. In the "k_1" row, we fix $k_2 = 10$ and in the "k_2" row, we fix $k_1 = 20$. We fix $\alpha = 0.2$, $\beta = 0.1$ and γ=0.05.

From Table 4, we could see that when fixing $k_2 = 10$, DGAE achieves the highest performance when k_1 is around 20. Based on the motivation of capturing same-area inconsistency, the top k_1 images consist of a part of images which are most similar to the query image in the POI, and a part of same-area inconsistent images in the POI based on the low-level feature similarity. We force both the most similar samples and the same-area inconsistent samples close to the query image. If k_1 is very small and less than 10, DGAE is only able to force the most similar images close and could not capture the same-area inconsistency. If k_1 is too large, DGAE may force some irrelevant images to be close, which causes the performance degrading. In the row "k_2", we find that when setting k_2 in the range from 5 to 25, DGAE is able to capture the easy-confusing inconsistency, in which images have similar low-level features but in different POIs. In the experiments, we set these parameters with cross-validation.

6.5.5 Discussion of kNN Model in Estimation

Table 5 shows the performance of k-NN model under k = 1 to 5. We report the top 1 prediction and the best of top

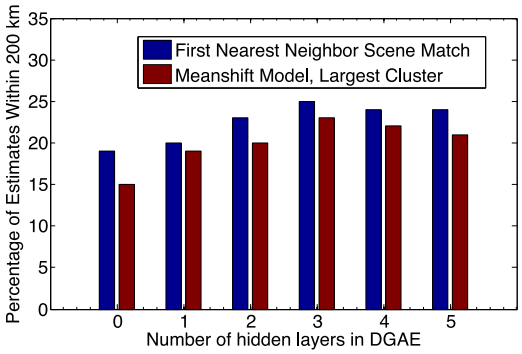


Fig. 8. Geo-localization performance across different layers of DGAE. Percentage of estimates within 200 km for each different settings of DGAE are shown. 1- 5 in x-axis presents the number of hidden layers used in DGAE. 0 in x-axis means only low-level features are used, which is present as the baseline.

TABLE 5
Geo-Localization Accuracy of Top-k Most Confident Predictions of DGAE

Performance (%)	city	region	country	continent
top 1	14.1	25.7	30.5	41.8
best top 2	14.9	26.3	32.5	50.6
best top 3	16.2	27.6	37.3	57.9
best top 4	17.3	28.4	40.7	63.8
best top 5	20.1	31.9	43.3	68.3

{2,3,4,5} predictions for each query. For example, when k= 3, first, we retrieve 3 most similar images as the query using feature representation learned with DGAE, and obtain the GPS of these three images. Second, we calculate the Err of the GPS of each image separately. Third, we pick the lowest Err as the performance. We could see that when comparing the best of the top-5 and top-1, the former one has roughly 1.5 times higher performance at all the levels.

6.6 Visualization of Estimated Results

Fig. 9 shows two examples of the GPS estimation results using LF[13], NFLL[14], AE and DGAE. Results of the best performance of each method according to cross-validation are presented. We could see that various approaches achieve different performance in these cities.

LF which only relays on low-level features performs the worst. In the first example, none of the top six results are within the same city by LF. There is only one image of the top six results within the same city. NFLL performs better than LF and is able to retrieve images with the scale and angle variations, such as the fourth result of NFLL. AE and DGAE achieve better performance than LF and NFLL. No less than two out of six results are within the same city. DGAE achieves the highest performance among four methods. Four out of six results are within the same city in these two examples.

We analyze that various approaches achieving different performance in these cities is mainly caused by whether these approaches could identify the discriminative features in these cities. Since LF only relays on low-level features such as color and texture, it may fail to learn discriminative features related to the image content. Clearly, images with similar low-level features may have very different content/high-level features. Compared to LF, NFLL adds SIFT descriptor to enhance the local feature representation based on SIFT points matches. We analyze that the SIFT feature based pair matching in NFLL plays a major role in excluding images with similar low-level features but irrelevant contents. Thus, NFLL is able to search images with same content but with scale and angle variations.

However, the variations of image contents in the close area are usually beyond scale and angle changes. AE and DGAE which learn mid/high-level feature representation achieve better performance than LF and NFLL. Compared to AE, we analyze that the attractive performance of DGAE are achieved by (1) the same-area constraint enhancing the rankings of the images within same area but may be with different low-level features; (2) the easy-confusing constraint reducing the rankings of images with similar low-level feature but in far away locations. For example, in the

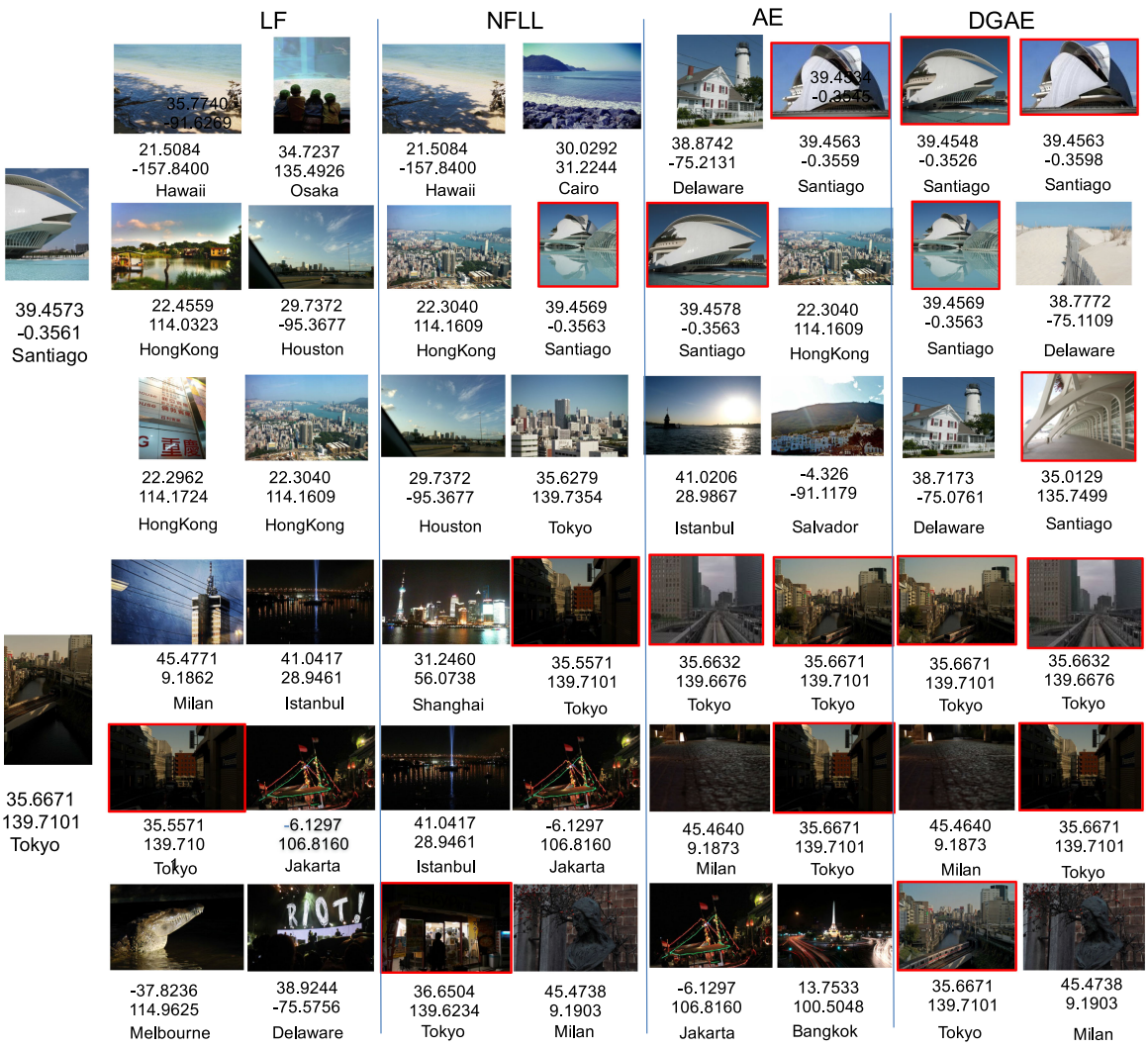


Fig. 9. Visualization of GPS estimation results by LF, NFLL, AE and DGAE. The left images are the input queries. The ground truth of geo-tag (i.e., latitude and longitude) is provided under each query image. For each query image, results of top 6 nearest neighbors are provided with scan order. To each result, the original geo-tag and area information are shown. Result images which are within the same city as ground truth of the query images are marked with red frame.

first example, when comparing AE and DGAE, we could see that the first result (incorrect) of AE is moved to the fifth result of DGAE, which may be due to the effectiveness of easy-confusing constraint that pushes out visually similar but geometrically far way images. The sixth result of DGAE does not appear in the top six results of AE and other two methods, which may be due to the effectiveness of same-area constraint that drags visually dis-similar but geometrically close images close.

6.7 Discussion of Computational Cost

In this section, we discuss the computational cost of both offline training and online estimation stages. In Table 6, we show results of the time cost of both a small subset of 27,369 images in New York City (named as "small") and a large dataset including all the areas (named as "large").

For offline training, there are three main time consuming steps. Step 1: mean-shift based geo-location clustering. Step 2: generating the pairwise graph for P and Q. Step 3: optimizing weights of DGAE. In Table 6, we name these three steps as "POI", "graph", "training". The fast version of graph generation in Section 4.2.1 is named as "graph(f)".

In the test stage, there mainly contains feature extraction and kNN search steps. 1-NN model is applied for both small and large dataset settings. For one image, it takes about 3 seconds on average to extract both low-level and high-level feature representations. The time cost of kNN search is related to the size of the dataset. The time cost of the test stage is named as "test".

When comparing "graph(f)" and "graph", we could see that in the large dataset, "graph(f)" is about 40 percent faster than "graph". It shows that the anchor based strategy greatly reduces the time cost of generating graph. In small dataset, the time difference is not large. The time which are saved in graph generating stage is not much larger than the additional time used for visual based clustering.

TABLE 6
Training Time of NCSCAE and LCSCAE

	POI	graph	graph(f)	training	test
small	1.20	0.33	0.26	0.89	0.001
large	5.33	20.22	12.16	18.32	0.03

The time costs are shown using the unit of hour.

6.8 Limitations and Future Works

In this section, we discuss the limitations of the current model and the potential solutions addressing these problems in the future.

In the current version, we mainly considered the spacial inconsistency constraints. Considering temporal correlation constraints with image geo-tagging would be a very interesting idea. Unfortunately, we found that a large portion of the timestamps in our current dataset are missing. We have also manually checked the quality of the timestamps, and found that there are many noises of the timestamps. For example, some images with timestamps in daytime may be night scenes. Thus, we are hesitated to directly merge the temporal information with DGAE model. We would like to explore the temporal correlation constraints in the future work.

Furthermore, currently, our model is working on a single machine. In the future, we would like to extend it to distributed systems to reduce the training time. It would be an interesting research topic of graph constrained deep learning models on distributed systems.

7 Conclusion

In this paper, we proposed a deep geo-constrained auto-encoder to extract deep feature presentation for automatic non-landmark GPS estimation. We addressed the key problems in GPS estimation, that images in same geo-area may be diverse while images with similar appearance may spread over the world. DGAE automatically minimizes the differences of images within the same geo-area, and meanwhile maximizes the differences of easy-confusing images which are allocated in different geo-areas but shared similar low-level features. In addition, we collected a new big dataset containing 664,720 images by Flickr open API. Extensive experimental results on the new dataset demonstrated the effectiveness of our DGAE based GPS estimation framework, which outperformed recent state-of-the-art works.

Acknowledgments

This research is supported in part by the NSF IIS award 1651902, ONR Young Investigator Award N00014-14-1-0484, and U.S. Army Research Office Award W911NF-17-1-0367.

References

[1] S. A. Ay, R. Zimmermann, and S. H. Kim, "Viewable scene modeling for geospatial video search," in *Proc. 16th ACM Int. Conf. Multimedia*, 2008, pp. 309–318.

[2] G. C. de Silva and K. Aizawa, "Retrieving multimedia travel stories using location data and spatial queries," in *Proc. 17th ACM Int. Conf. Multimedia*, 2009, pp. 785–788.

[3] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 72–79.

[4] W.-C. Chen, A. Battestini, N. Gelfand, and V. Setlur, "Visual summaries of popular landmarks from community photo collections," in *Conf. Rec. 43rd Asilomar Conf. Signals Syst. Comput.*, 2009, pp. 1248–1255.

[5] S. Jiang, X. Qian, T. Mei, and Y. Fu, "Personalized travel sequence recommendation on multi-source big social media," *IEEE Trans. Big Data*, vol. 2, no. 1, pp. 43–56, Mar. 2016.

[6] S. Jiang, X. Qian, J. Shen, Y. Fu, and T. Mei, "Author topic model-based collaborative filtering for personalized poi recommendations," *IEEE Trans. Multimedia*, vol. 17, no. 6, pp. 907–918, Jun. 2015.

[7] D. M. Chen, et al., "City-scale landmark identification on mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 737–744.

[8] Y. Avrithis, Y. Kalantidis, G. Tolias, and E. Spyrou, "Retrieving landmark and non-landmark images from community photo collections," in *Proc. 18th ACM Int. Conf. Multimedia*, 2010, pp. 153–162.

[9] Y.-T. Zheng, et al., "Tour the world: Building a web-scale landmark recognition engine," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 1085–1092.

[10] T.-Y. Lin, S. Belongie, and J. Hays, "Cross-view image geolocalization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 891–898.

[11] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5007–5015.

[12] S. Workman, R. Souvenir, and N. Jacobs, "Wide-area image geolocalization with aerial reference imagery," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 3961–3969.

[13] J. Hays and A. Efros, "IM2GPS: Estimating geographic information from AB single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[14] J. Hays and A. A. Efros, "Large-scale image geolocalization," in *Multimodal Location Estimation of Videos and Images*. Berlin, Germany: Springer, 2015, pp. 41–62.

[15] D. Kit, Y. Kong, and Y. Fu, "Lasom: Location aware self-organizing map for discovering similar and unique visual features of geographical locations," in *Proc. Int. Joint Conf. Neural Netw.*, 2014, pp. 263–270.

[16] D. Joshi, A. Gallagher, J. Yu, and J. Luo, "Exploring user image tags for geo-location inference," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2010, pp. 5598–5601.

[17] L. Cao, J. Luo, A. Gallagher, X. Jin, J. Han, and T. Huang, "A worldwide tourism recommendation system based on geotaggedweb photos," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2010, pp. 2274–2277.

[18] E. Kalogerakis, O. Vesselova, J. Hays, A. Efros, and A. Hertzmann, "Image sequence geolocation with human travel priors," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 253–260.

[19] D. J. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg, "Mapping the world's photos," in *Proc. 18th Int. Conf. World Wide Web*, 2009, pp. 761–770.

[20] P. Serdyukov, V. Murdock, and R. van Zwol, "Placing flickr photos on a map," in *Proc. 32nd Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval*, 2009, pp. 484–491.

[21] M. Cristani, A. Perina, U. Castellani, and V. Murino, " Geo-located image categorization and location recognition," *Pattern Recog. Image Anal.*, vol. 19, no. 2, pp. 245–252, 2009.

[22] S. Cao and N. Snavely, "Graph-based discriminative learning for location recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 700–707.

[23] Y.-C. Song, Y.-D. Zhang, J. Cao, T. Xia, W. Liu, and J.-T. Li, "Web video geolocation by geotagged social resources," *IEEE Trans. Multimedia*, vol. 14, no. 2, pp. 456–470, Feb. 2012.

[24] T. Weyand, I. Kostrikov, and J. Philbin, "Planet-photo geolocation with convolutional neural networks," *Eur. Conf. Comput. Vis.*, pp. 37–55, 2016.

[25] Y.-H. Kuo, W.-Y. Lee, W. H. Hsu, and W.-H. Cheng, "Augmenting mobile city-view image retrieval with context-rich user-contributed photos," in *Proc. 19th ACM Multimedia*, 2011, pp. 687–690.

[26] A. Bergamo, S. N. Sinha, and L. Torresani, "Leveraging structure from motion to learn discriminative codebooks for scalable landmark classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 763–770.

[27] Y. Li, D. J. Crandall, and D. P. Huttenlocher, "Landmark classification in large-scale image collections," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 1957–1964.

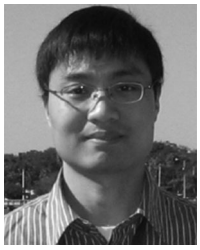
[28] J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, and A. Oliva, "Sun database: Exploring a large collection of scene categories," *Int. J. Comput. Vision*, vol. 119, no. 1, pp. 3–22, 2016.

[29] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

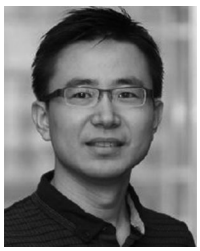
[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[31] D. Kit, Y. Kong, and Y. Fu, "Efficient image geotagging using large databases," *IEEE Trans. Big Data*, vol. 2, no. 4, pp. 325–338, Dec. 2016.
[32] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.
[33] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura, "Travel route recommendation using geotags in photo sharing sites," in *Proc. 19th ACM Int. Conf. Inform. Knowl. Manag.*, 2010, pp. 579–588.
[34] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, 2010.
[35] Y. Bengio, L. Yao, G. Alain, and P. Vincent, "Generalized denoising auto-encoders as generative models," in *Proc. Int. Conf. Adv. Neural Inform. Process. Syst.*, 2013, pp. 899–907.
[36] M. Chen, K. Q. Weinberger, F. Sha, and Y. Bengio, "Marginalized denoising auto-encoders for nonlinear representations," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 1476–1484.
[37] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.
[38] M. Kan, S. Shan, H. Chang, and X. Chen, "Stacked progressive auto-encoders (SPAE) for face recognition across poses," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1883–1890.
[39] Y. Bengio, "Learning deep architectures for WI," *Found. Trends® Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.
[40] M. Wang, W. Fu, S. Hao, D. Tao, and X. Wu, "Scalable semi-supervised learning by efficient anchor graph regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1864–1877, Jul. 2016.
[41] M. Wang, W. Fu, S. Hao, H. Liu, and X. Wu, "Learning on big graph: Label inference and regularization with anchor hierarchy," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 5, pp. 1101–1114, May 2017.
[42] W. Liu, J. He, and S.-F. Chang, "Large graph construction for scalable semi-supervised learning," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 679–686.
[43] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 40–51, Jan. 2007.
[44] J. Hu, J. Lu, and Y.-P. Tan, "Deep transfer metric learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 325–333.
[45] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive Model.*, vol. 5, no. 3, p. 1, 1988.
[46] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. Cambridge, MA, USA: The MIT Press, 2006.
[47] J. Li, X. Qian, Y. Y. Tang, L. Yang, and T. Mei, "GPS estimation for places of interest from social users' uploaded photos," *IEEE Trans. Multimedia*, vol. 15, no. 8, pp. 2058–2071, Dec. 2013.
[48] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks," *J. Mach. Learn. Res.*, vol. 10, pp. 1–40, Jan. 2009.
[49] S. Jiang, M. Shao, C. Jia, and Y. Fu, "Consensus style centralizing auto-encoder for weak style classification," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2016, pp. 1223–1229.

Shuhui Jiang received the BS and the MS degrees from the Xi'an Jiaotong University, Xi'an, China, in 2007 and 2011, respectively. She is now pursuing the PhD degree in the School of Electrical and Computer Engineering, Northeastern University, Boston. She was the recipient of the Dean's Fellowship of Northeastern University from 2014. She is interested in machine learning, multimedia and computer vision. She has served as a reviewer of the IEEE journals: The *IEEE Transactions on Neural Networks and Learning Systems* etc. She was a research intern with Adobe research lab, San Jose, US, in summer 2016.



Yu Kong received BEng degree in automation from the Anhui University, and the PhD degree in computer science from the Beijing Institute of Technology, China, in 2006 and 2012. He was a visiting student in the National Laboratory of Pattern Recognition (NLPR), Chinese Academy of Science from 2007 to 2009, and a visiting scholar in the Department of Computer Science and Engineering, State University of New York, Buffalo, in 2012. He is now a postdoctoral research associate in the Electrical and Computer Engineering, Northeastern University, Boston, MA. His research interests include computer vision, social media analytics, and machine learning. He has served as a peer reviewer of the IEEE transactions including the *IEEE Transactions on Pattern Analysis and Machine Intelligence* and the *IEEE Transactions on Cybernetics*. He is a member of the IEEE.



Yun Fu (S'07-M'08-SM'11) received the BEng degree in information engineering and the MEng degree in pattern recognition and intelligence systems from the Xi'an Jiaotong University, China, respectively, and the MS degree in statistics and the PhD degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, respectively. He is an interdisciplinary faculty member affiliated in the College of Engineering and the College of Computer and Information Science, Northeastern University since 2012. His research interests include the machine learning, computational intelligence, big data mining, computer vision, pattern recognition, and cyber-physical systems. He has extensive publications in leading journals, books/book chapters and international conferences/workshops. He serves as associate editor, chairs, PC member and reviewer of many top journals and international conferences/workshops. He received seven Prestigious Young Investigator Awards from NAE, ONR, ARO, IEEE, INNS, UIUC, Grainger Foundation; nine seven Best Paper Awards from the IEEE, ACM, IAPR, SPIE, SIAM; three major Industrial Research Awards from Google, Samsung, and Adobe, etc. He is currently an associate editor of the *IEEE Transactions on Neural Networks and Leaning Systems (TNNLS)*. He is fellow of IAPR, a Lifetime senior member of the ACM and SPIE, Lifetime member of AAAI, OSA, and Institute of Mathematical Statistics, member of Global Young Academy (GYA), INNS and Beckman Graduate Fellow during 2007-2008. He is a senior member of the IEEE.