Automatic image annotation using semi-supervised generative modeling

S. Hamid Amiri, Mansour Jamzad *

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

1. Introduction

The growth of digital image datasets and photo sharing communities on the Internet makes it necessary to provide a proper mechanism for searching the images in large collections. Content-based image retrieval (CBIR) focuses on the problem of searching images based on their content. The first generation of CBIR was based on query-by-example (QBE) paradigm [1], in which the CBIR system retrieves most visually similar images to a query image given by the user. The semantic gap between low level features and high level concepts is a fundamental problem in QBE systems. To bridge the semantic gap, many systems use the relevance feedback approach [2,3] to incorporate user knowledge into the retrieval process. Besides these query-based search strategies, the new generation focuses on development of an automatic system that semantically describes the content of an image. In this approach, a set of semantic labels are assigned to each image to describe its content. Then, a system is developed to train a model for the relation between visual features and tags of images. This formulation, called automatic image annotation or linguistic indexing, allows the system to assign words to every new test image. Furthermore, the retrieval procedure could be performed based on input texts provided by the user. To bridge the semantic gap in an annotation system, some methods utilized active learning [4] to exploit the user knowledge in developing the annotation system and thus to reduce the required supervision. In what follows, we briefly overview main features of the prior studies.

1.1. Related works

There has been a great effort to design an annotation system using statistical learning. According to Ref. [5], one strategy for statistical annotation is unsupervised labeling [6,7] which estimates the joint density of visual features and words by an unsupervised learning algorithm. These methods introduce a hidden variable and assume that features and words are independent given the value of hidden variable. Another formulation for statistical annotation is supervised multi-class labeling (SML) [5,8] that estimates a conditional distribution for each semantic class to determine probability of a feature vector given the semantic label. Regardless of learning strategy, training dataset plays a major role in developing annotation systems. Manual assignment of too many words to a large number of images in the dataset is highly time consuming and labor intensive. On the other hand, a classifier may have poor generalization when some semantic labels have a few images. It seems essential to develop an annotation system that depends on a small number of labeled images in the training phase. Although it is difficult to prepare an annotated dataset, one could easily obtain unlabeled images in large quantity (e.g., using photo sharing communities on the Internet like Flicker and ImageNet). This large amount of pictures motivates image annotation systems to increase their generalization by incorporating unlabeled images into
the training phase. To reach this aim, semi-supervised learning (SSL) [9] is emerging in machine learning community. Semi-supervised generative models [10,11] and graph-based methods [12] are two main classes of the SSL that have recently received lots of interests, especially in image annotation [13–20].

A large number of semi-supervised annotations utilize graph-based learning to infer tags of unlabeled images. The main challenging issue in these methods is graph construction. Liu et al. [13] proposed a method called nearest spanning chain (NSC) which constructs the learning graph using chain-wise statistical information instead of the traditional pair-wise similarities. Besides constructing multiple NSCs that is computationally expensive, they do not show how the number of labeled images affects the annotation results. The authors in Ref. [14] focused on graph construction in the presence of noisy annotations. To this end, by solving an optimization problem, a sparse graph is constructed and the training process is modified to handle noisy annotations. A disadvantage of this approach is that edges of the sparse graph are considered to be a subset of edges in a kNN graph. The method in Ref. [15] constructs a bi-relational graph that comprises both visual features and semantic labels of images. By propagating the words of annotated images over bi-relational graph, annotations of unlabeled images are extracted.

To further refine the annotation results, there are some works that combine graph-based learning with other techniques. Tang et al. [16] proposed a method to combine graph-based learning with multiple instance learning [17] for image annotation. After constructing two graphs based on multiple and single instance representations, the graphs are integrated into a unified graph for the learning process. This method utilizes a simple weighted-sum rule for integration. It suffers from early fusion problems in multimodal representation [18]. Shao et al. [19] presented a framework to combine graph-based learning with a probabilistic model for learning latent topics of images. In this framework, a hidden variable is associated to each image and probabilities of hidden variables are considered to reside on a manifold. This approach suffers from the limitations of unsupervised labeling discussed in Ref. [5]. Zhu et al. [20] proposed a technique which utilizes graph-based learning to refine the candidate annotations obtained from the progressive relevance-based method [21]. Since the candidate annotations are obtained by propagating words from labeled images to unlabeled ones, noisy annotations will highly degrade the performance of this approach.

In addition to the above limitations, there are two main problems that should be considered in graph-based image annotation. First, these methods are transductive and can only predict labels for specific unlabeled samples observed in the training phase. To annotate a new test image, we must add the image to the unlabeled set and run the training phase again. The second problem is related to the required memory and time complexity of graph-based approaches. The primary bottleneck of these approaches comes from the complexity of handling the adjacency matrix of graph which super-linearly grows by increasing the number of images.

It is shown that annotation could be performed in a short time when generative models are utilized in SML formulation [8]. Additionally, generative models are inductive and can predict the label of every sample in the feature space. Thus, in this work, we focus on the semi-supervised generative models for image annotation. We follow PSU protocol [5] for developing the system which is previously utilized in a supervised system called ALIPR [8].

1.2. Overview of ALIPR

ALIPR utilizes Corel60k [7] dataset whose images are organized based on PSU protocol [5]. This protocol assumes that images are divided into distinct categories or “concepts”. Each concept contains a set of words describing the entire images in that concept even though some of these words do not occur in each individual image. It is possible that two different concepts could share some words in their descriptions. Corel60k is comprised of 599 concepts with about 100 images in each concept. This dataset also includes total of 417 distinct words.

With the above structure, ALIPR constructs a generative model for each concept as follows:

1. Extracting the color and texture signatures of images.
2. Partitioning images of a concept using a new clustering algorithm named D2-clustering and extracting a prototype for each cluster.
3. Computing the distance between each prototype and the signatures assigned to it.
4. Fitting Gamma distribution into each cluster based on the distances in that cluster.
5. Combining the distribution components to construct a mixture model for each concept.

The advantage of ALIPR over the prior statistical systems such as [5–7] was its higher speed in annotation procedure.

1.3. Contributions of this paper

The current study targets to develop a semi-supervised annotation system to include unlabeled images in concept modeling. To this end, it covers the following contributions:

- We propose a new feature extraction strategy that obtains color and texture signatures in less time than ALIPR. Besides, the new features provide more discriminative property.
- We embedded spectral clustering in the prototype extraction to overcome limitations of ALIPR. Indeed, D2-clustering in ALIPR is a divisive hierarchical clustering and has two limitations. First, since a set of linear optimization problems with a large number of variables must be solved in each level of clustering, the prototype extraction is computationally expensive. Second, the division process utilizes a greedy strategy which could not lead to well-structured clusters.
- To use unlabeled images in the training phase, the formulation of prototype extraction is modified by assigning a weight to each signature that indicates the membership degree of the signature to one concept.
- Given an initial model for each concept, we incorporate unlabeled images into the training phase to improve models through modifying parameters of the clusters. Thus, a new parameter estimation method is presented which utilizes the unlabeled images.

The last two contributions provide semi-supervised annotation and are considered as main contributions of the proposed solution. On the other hand, the first two contributions are not the main contributions but are mandatory for achieving good efficiency in the annotation system.

In our approach, we assume that training dataset follows PSU protocol. However, in our experiments, we will also discuss how our approach can be applied to datasets that are not organized based on PSU protocol. In fact, the PSU protocol assumption in the training phase does not weaken the applicability of our approach to other datasets.

1 An early version of supervised framework [24] explains the feature extraction and clustering phases.
The reminder of this paper is organized as follow. Feature extraction is discussed in Section 2. In Section 3, we focus on the problem of prototype extraction for a set of image signatures. Details of the training phase are discussed in Section 4. The experimental results are given in Section 5. Finally, we conclude our work in Section 6.

2. Feature extraction

To describe the content of images, color and texture features are extracted in this work. We refer to color and texture descriptors of an image as its signature. To form a signature, color and texture segmentations are applied to the given input image. A set of two distinct clustering algorithms as described below.

To segment an image based on color, it is mapped from RGB color space to Lab space. Then, we apply Meanshift [22] segmentation followed by an agglomerative hierarchical clustering [23] to Lab components of pixels. The detail of this algorithm is discussed in Ref. [24].

To perform texture segmentation, the gray-scale version of input image obtained by L component is divided into distinct blocks with size of 16 × 16 pixel. Then, Gabor features [25] are extracted from each block and a clustering algorithm is applied to these feature vectors for texture-based segmentation. The detail of this algorithm is given in Ref. [24].

Suppose that the above procedure provides m segments for an input image. The centroid of each segment is defined by averaging on its feature vectors. To extract a descriptor from segmentation output, a discrete distribution is defined as \((\nu_1, p_1), (\nu_2, p_2), \ldots, (\nu_m, p_m)\), where \(\nu_i\) is the centroid of the ith segment and \(p_i\) denotes its probability which is estimated by the percentage of pixels associated to the ith segment. Since each image is segmented using color and texture separately, we get two discrete distributions to form the image signature.

To compute the distance between two image signatures, we need a metric defined between two discrete distributions. This way, Mallows distance [26] is utilized to calculate the distance of two discrete distributions \(d_1\) and \(d_2\) defined as

\[
D^2(d_1, d_2) = \min_{W} \sum_{i=1}^{m} w_i \| \nu_i - \nu_i' \|^2
\]

where \(w_i\) corresponds to joint probability of \(\nu_i\) and \(\nu_i'\). Because of the marginal distributions, we must add the following constraints to the above problem:

\[
\sum_{j=1}^{m} w_{ij} = p_{ij} \quad \sum_{i=1}^{m} w_{ij} = p_j \quad j = 1, \ldots, m_2 \quad i = 1, \ldots, m_1
\]

The objective function \(D^2\) in Eq. (1) and constraints in Eq. (2) are linear with respect to the set of optimization variables \(W\), forming a problem with linear programming (LP) solution.

Note that an image signature is comprised of two separate discrete distributions for color and texture. Therefore, the total distance between pair of signatures \(I_i\) and \(I_j\) is calculated by the summation on the squared Mallows distance between their individual discrete distributions as follows:

\[
\hat{D}(I_i, I_j) = \frac{d}{\sum_{i=1}^{d} \lambda_i D^2(I_i, I_j)}
\]

where \(d\) is the number of distributions for a signature and \(\lambda_i\) denotes the \(i\)th distribution for \(I_i\). In our experiments, a signature consists of color and texture parts, therefore we set \(d=2\). \(\lambda_i\) is a fixed weight used for normalization of distance values for color and texture parts. To determine these weights, we select a set of images with similar contents and compute the Mallows distance among all pairs of their color (texture) distributions. Then, we statistically analyze the distance values to specify the weights \(\lambda_1\) and \(\lambda_2\) in such a way that color and texture parts have equal importance in computing the total distance \(\hat{D}\) in Eq. (3). The details of this strategy are given in Section 5.1.

Finally, the feature extraction methodology in this paper can be distinguished from the one in ALIPR as described here. While both ALIPR and the proposed system extract color components in Lab space, we take a different approach for color segmentation (i.e. a two-step clustering approach [24]). To obtain texture signature, ALIPR extracts texture descriptors from wavelet domain and performs K-means clustering on feature vectors. In contrast, we apply Gabor transform to images and utilize a graph-based clustering approach [24] for texture segmentation. Gabor has the advantage of providing more details for an image in transform domain.

3. Prototype extraction

Assume that a set of image signatures is given and a weight is assigned to each signature representing its importance. The goal is to extract one prototype as the representative of the set of signatures. This problem is similar to the update step in K-means clustering which calculates the centroid of a cluster.

We could formulate the extraction of prototype \(a_i\) for a cluster \(C_k\) as follows:

\[
a_i = \arg \min_{\alpha \in \Omega} \sum_{k \in C_k} \rho_k D^2(I_i, a_j)
\]

where \(\Omega = \Omega_1 \times \Omega_2\) defines the space of prototypes such that \(\Omega_k\) \((k=1,2)\) indicates the space of \(k\)th discrete distribution in the image signature. Also, \(\rho_1\) is the weight of image signature \(I_i\). Since the prototype must have the same structure as image signatures, it contains two discrete distributions for color and texture parts, as well. Substituting Eq. (3) in (4), we have

\[
a_i = \arg \min_{\alpha \in \Omega} \sum_{k \in C_k} \rho_k \sum_{j \in \Omega^1} \lambda_j D^2(I_i, a_j) = \sum_{i=1}^{\frac{m_1}{2}} \lambda_i \sum_{j=1}^{\frac{m_2}{2}} \sum_{i=1}^{\frac{m_1}{2}} \rho_k D^2(I_i, a_j)
\]

Where \(I_i = (\nu_{i1}, p_{i1}), (\nu_{i2}, p_{i2}), \ldots, (\nu_{im_1}, p_{im_1})\) is the \(i\)th discrete distribution in \(I_i\). The above equation indicates that optimization must be performed for color and texture parts of the signatures, separately. By applying Mallows distance in Ref. (1) to the above problem, we have to solve the optimization problem in Eq. (6) to determine \(a_i\):\(=(\alpha_i, \beta_i, \ldots, (\alpha_{m_1}, \beta_{m_1})\):

\[
\min_{\alpha_1, \ldots, \alpha_{m_1}} \sum_{k=1}^{m_1} \rho_k D^2(I_i, a_j) = \sum_{k=1}^{m_1} \sum_{j=1}^{m_1} \sum_{i=1}^{m_2} w_{ij} p_{ij}^2 - \sum_{i=1}^{m_1} w_{ij} q_{ij}^2 - \sum_{i=1}^{m_1} w_{ij} \alpha_i^2 - \sum_{i=1}^{m_1} w_{ij} \beta_i^2
\]

such that:

\[
\sum_{k=1}^{m_1} \rho_k q_{ij} = 0, \quad \sum_{j=1}^{m_1} \rho_k w_{ij} \alpha_i = 0, \quad \sum_{j=1}^{m_1} \rho_k w_{ij} \beta_i = 0
\]

To have a solution for the above minimization problem with variables \(w_{ij}, q_{ij}\) and \(q_{ij}\), a two-step iterative algorithm is performed. At first step, we assume \(q_{ij}\)’s are fixed and solve the optimization for \(w_{ij}\)’s and \(q_{ij}\)’s. As the constraints and objective function are linear in this step, we use the linear programming for optimization. In the second step, given \(w_{ij}\)’s and \(q_{ij}\)’s, we update \(q_{ij}\). If we compute the derivative of the objective function in Eq. (6) with respect to \(q_{ij}\) and set it to zero, we have

\[
q_{ij} = \frac{\sum_{k=1}^{m_1} \rho_k w_{ij} \alpha_i \beta_i}{\sum_{k=1}^{m_1} \rho_k w_{ij}}
\]
After initializing the prototype, two steps are iterated until the change in the objective function becomes less than a threshold. The value of this threshold will be defined in Section 5.1. Initialization of prototype $a_{ij}$ is also explained in Section 4.1.

The above procedure for prototype extraction will be used during the training process. Thus, based on our statistical learning framework, we can compute the probability of belonging an image signature to a cluster. This probability is considered as the weight of that signature during the prototype extraction. Therefore, the weight $\rho_i$ is adaptively specified during the training process. The details are given in Section 4.3.2.

It should be noted that the above procedure is a modified version of prototype extraction step in D2-clustering [8]. The main difference is that we consider a weight for each signature. Before applying prototype extraction, images must be clustered into separate groups. D2-clustering groups similar signatures together through a divisive hierarchical clustering. After calculating a prototype for an image cluster, D2-clustering splits that cluster and performs prototype extraction step again. This greedy strategy leads to suboptimal clusters and also makes clustering algorithm computationally expensive. We extract groups of similar signatures by spectral clustering that is introduced in Section 4. We call the above algorithm as weighted D2-clustering indicating weighted signatures are used for clustering.

4. Training

We assume that the training images are organized based on PSU protocol as discussed in Section 1.2. The objective of training phase is to fit a mixture model into each concept. The model calculates the probability that an image appears at the concept. Since the design of annotation system is performed based on the semi-supervised learning, we employ both the unlabeled and labeled images of a concept to construct its model. Note that unlabeled images are shared among all concepts in the training phase.

The mixture model of a concept is obtained in two major steps. At first, an initial model is constructed using the labeled images of that concept. Then, the parameters of models are updated by incorporating unlabeled images.

4.1. Initial modeling

To construct the initial model of a specific concept, we follow three steps: (1) Clustering the labeled images of the concept; (2) calculating one prototype for each cluster; and (3) fitting a mixture model based on the prototypes in the concept.

To cluster the images of one concept, we utilized the self-tuning spectral clustering algorithm in Ref. [27]. This algorithm is implemented by a similarity graph $G=(V,W)$ whose nodes correspond to labeled image signatures of the concept. The distance of nodes $i$ and $j$ is also defined as follows:

$$d(i,j) = D^{(1)}(i,j) / \sigma_c + (D^{(2)}(i,j) / \sigma_t) \quad (8)$$

where $D^{(1)}$ and $D^{(2)}$ compute the Mallows distances between color and texture signatures, respectively. To normalize these distances before combining, we divide them by $\sigma_c$ and $\sigma_t$ the standard deviations of color and texture distances between all images.

To form the edge matrix $W$, each node is connected to its $k$ nearest neighbors on the graph. The weights of these edges are determined using the Gaussian function:

$$w_{ij} = \exp(-d(i,j) / \sigma^2) \quad (9)$$

where $\sigma$ is the scaling parameter determined automatically using local scaling approach [27].

We run the self-tuning spectral clustering [27] on the constructed graph to obtain the clusters of nodes. The advantage of this algorithm is automatic selection of the number of clusters.

Our experiments show that small values for $k$ in kNN approach work well for clustering. Thus, we set $k=3$ in the experiments. As a sample of clustering algorithm, Fig. 1 shows the output for concept “autumn” in Corel60k dataset when the number of labeled images is 80.

After clustering the labeled images of one concept, we must determine a prototype for each cluster of signatures. To this end, we employ the prototype extraction algorithm in Section 3. In this algorithm, $\rho_i$ (weight of image signature $l_i$) indicates the prior probability that $l_i$ belongs to a concept. Since a labeled image belongs to a certain concept, we set its weight to one in the prototype extraction algorithm.

As stated in Section 3, the prototype of each cluster is determined in an iterative strategy. To initialize the prototype of a typical cluster $C_k$ we compute the closeness centrality [28] for each signature in the cluster. Then, the signature with maximum closeness centrality is selected as the initial prototype. This process is discussed in details in Ref. [24]. It should be noted that the discrete distribution $\alpha_{k,t}$ (see Section 3) in the initial prototype is forced to follow uniform distribution.

After determining the prototypes of one concept, a mixture model is constructed for that concept. Similar to ALIPR, we design the kernel of mixtures based on Mallows distance between each image signature and its nearest prototype. It has been shown that these distances follow Gamma distribution with two parameters $b$ (scale) and $s$ (shape) [8]:

$$f(u) = (u/b)^{b-1} e^{-u/b} / b^s \Gamma(s), \quad u \geq 0 \quad (10)$$

where $u$ indicates the distance sample and $\Gamma(.)$ is Gamma function.

Thus, we could fit a Gamma distribution into each cluster based on Mallows distances between the prototype and signatures of that cluster. This results in Eq. (11) for the probability of belonging one image signature $l$ to a cluster with prototype $\alpha_k$ [8]:

$$p(l|\alpha_k) = (1/\sqrt{b}) b^{s/2} \exp(-\bar{D}(l, \alpha_k)/b) \quad (11)$$

Here, $b_k$ and $s$ are the parameters of Gamma distribution.

Assuming a prototype as the centroid of a mixture component, the overall model of the $k$th concept with $m$ prototypes is obtained by combining the distributions around the prototypes:

$$\phi(l|\mu_k) = \sum_{\eta=1}^{m} \alpha_{k,\eta} (1/\sqrt{s b_k})^{s/2} \exp(-\bar{D}(l, \mu_k)/b) \quad (12)$$

Fig. 1. The output of clustering for concept “autumn”. Images with similar cluster are enclosed by dashed lines.
where $\mu_k$ is the model of $k$th concept. Also, $\omega_l$’s are the prior probabilities for clusters $q=1,..,m$ in $k$th concept with the constraint $\sum_m \omega_q = 1$. These priors are estimated by the percentage of signatures assigned to the prototype $\omega_q$. It should be noted that a common parameter $s$ is used for all component distributions in all concepts. As this parameter specifies the shape of Gamma distribution, all component distributions have the same shape.

To estimate the parameters of components in the initial models, the maximum likelihood (ML) method is used. We will discuss about the objective function of estimator in the next section. Also, we will explain how to estimate the parameters of initial models in Section 4.4.

### 4.2. Objective function for parameter estimation

Suppose that $M_q$ represents the number of prototypes for $k$th concept, $M = \sum_q M_q$ is the total number of clusters, and, $C_j$ ($j=1,...,M$) indicates the $j$th cluster with prototype $\alpha_j$. As defined in Eq. (12), the generative model of one concept is based on Mallows distances between image signatures and prototypes. Therefore, the likelihood function is expressed in term of Mallows distances between signatures and their nearest prototypes.

We assume that every signature is confined to fall into one cluster. If $Concept_{l_i}$ is the concept index of signature $l_i$, we can convert the sample $(l_i, Concept_{l_i})$ into a unique sample $(u_i, z_i)$ where $z_i$ is the index of a cluster that contains $l_i$. In fact, $z_i$ is a random variable which takes values 1 to $M$, where $M$ is the total number of clusters. Moreover, $u_i$ represents Mallows distance between $l_i$ and the prototype of cluster with index $z_i$. Considering $\theta = (\alpha_0, \beta_0, s)$ as the parameters of cluster $C_l$, the likelihood function for $N$ signatures is defined as follows:

$$L(\theta; u_i, z_i) = \prod_{i=1}^{N} \prod_{j=1}^{M} \theta_j^{z_i j} (1-\theta_j)^{1-z_i j} f(u_i | \theta_j)$$

where $\theta = (\alpha_0, \beta_0, s)$ is the set of parameters for $M$ clusters. The sets $\mathcal{Z} = \{z_1, ..., z_N\}$ and $\mathcal{U} = \{u_1, ..., u_N\}$ indicate cluster indices and Mallows distances between $N$ signatures and their corresponding prototypes, respectively. Additionally, $f(u_i | \theta_j)$ is defined in Eq. (10) and $\theta_j$ denotes the indicator function that returns one if its input expression is true and zero otherwise.

Eq. (13) could be rewritten as follows:

$$L(\theta; u_i, z_i) = \exp \left( \sum_{i=1}^{N} \sum_{j=1}^{M} (z_i \cdot j) \log f(u_i | \theta_j) \right)$$

$$= \log L(\theta; u_i, z_i) = \sum_{i=1}^{N} \sum_{j=1}^{M} f(u_i | \theta_j)$$

This function is regarded as the objective function for parameter estimation. Given the sets $\mathcal{Z}$ and $\mathcal{U}$, the ML estimation of $\theta$ is obtained by maximizing $\log L(\theta; u_i, z_i)$.

### 4.3. Incorporating unlabeled images

After constructing the initial models of concepts, we incorporate unlabeled images to update the parameters of component distributions of the generative models. As discussed in previous section, the parameters are estimated using samples $(u_i, z_i)$, which stem from complete observations $(l_i, Concept_{l_i})$. We will have a complete observation on an image signature $l_i$ if we know to which concept it belongs. Unfortunately, the concepts of unlabeled images are unknown which leads to incomplete observations. We must convert them to complete observations to incorporate unlabeled images in parameter estimation.

Let $\mathcal{D}_l = \{l_1, Concept_{l_1}, ..., l_u, Concept_{l_u}\}$ and $\mathcal{D}_u = \{l_{u+1}, ..., l_{u+u}\}$ to be the labeled and unlabeled image sets where $l$ and $u$ indicate the number of labeled and unlabeled images, respectively. We arrange a vector $\Pi_l$ for each image signature $l_i$ such that its dimension is equal to the number of concepts. Each element of this vector demonstrates the membership degree of a signature to its corresponding concept. If the labeled signature $l_i$ is assigned to $k$th concept ($Concept_{l_i} = k$), then the $k$th component of $\Pi_l$ will be equal to one, otherwise its value is zero. To form this vector for unlabeled images, we use the current models of concepts to calculate the membership degrees. Therefore for $T$ concepts, the elements of $\Pi_l$ are defined as follows:

$$\Pi_l(k) = \frac{p_{\alpha_k} \cdot l_i \in \mathcal{D}_u}{\sum_{l \in \mathcal{D}_u} p_{\alpha_l}}, \quad l_i \in \mathcal{D}_u \quad \forall k = 1, .., T$$

where $p_k$ is the prior probability for $k$th concept with uniform distribution. Also, $l(.)$ is the indicator function and $\mu_k$ and $\mu_l$ indicate the generative model for concepts $k$ and $l$, respectively.

The concept index of an unlabeled image is regarded as a hidden variable. Since each sample $(l_i, Concept_{l_i})$ is converted to a unique sample $(u_i, z_i)$, the variables $z_i$s are also hidden. To find the ML estimation of parameters in the presence of hidden variables, we use a modified EM algorithm. This algorithm alternates between $E$-step and $M$-step as will be explained.

#### 4.3.1. Calculating the posterior probabilities

In $E$-step, we compute the expected value of log-likelihood function in Eq. (14) with respect to conditional distribution of $\mathcal{Z}$ given $\mathcal{U}$ and $\theta^{(t)}$ as current estimation of parameters in iteration $t$. This yields the following equation:

$$Q(\theta) = E_{\mathcal{Z} | \mathcal{U}^{(t)}, \theta^{(t)}} \log L(\theta; u_i, z_i)$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{M} E[\mathcal{Z} | \mathcal{U}^{(t)}, \theta^{(t)}] \log f(u_i | \theta_j)$$

where $E[\mathcal{Z} | \mathcal{U}^{(t)}, \theta^{(t)}]$ is defined as follows:

$$E[\mathcal{Z} | \mathcal{U}^{(t)}, \theta^{(t)}] = p(z_i = j)$$

The term $p(z_i = j)$, in this equation, is the probability that image signature $l_i$ belongs to the cluster $C_j$. To determine this probability, we use the fact that each signature is assigned to the closest prototype in a concept. Let $C_j$ be a cluster in the $k$th concept. If it contains the closest prototype to signature $l_i$, we set $p(z_i = j) = \Pi_l(k)$, the prior probability of concept as defined in Eq. (15). For other clusters in the $k$th concept, $p(z_i = j) = 0$. Therefore, we have the following equation to compute $p(z_i = j)$:

$$p(z_i = j) = \Pi_l(k) \times 1(C_j = Nearest_t(l_i))$$

where $Nearest_t(l_i)$ is a cluster in $k$th concept that contains the closest prototype to $l_i$. Obviously, if $l_i$ is a labeled image with concept index $k$ ($Concept_{l_i} = k$), then $p(z_i = j) = 1$ for the closest prototype of $k$th concept. Otherwise $p(z_i = j) = 0$. To sum up, $E$-step calculates the probabilities $p(z_i = j)$ using Eq. (18) to determine the prior probabilities for signatures.

#### 4.3.2. Updating the parameters of clusters

Based on the labeled and unlabeled images and their prior probabilities, $M$-step updates the parameters of each cluster. We explain the details of this step for a typical cluster $C_j$ in the following. For other clusters, the update process is performed in an identical manner.

At first, we utilize weighted D2-clustering algorithm described in Section 3 to update the prototype of cluster $C_j$ using labeled and unlabeled images assigned to the cluster. To run the clustering algorithm, we consider the current prototype as the initial prototype and follow the iterations discussed in Section 3. Moreover, the weights of signatures are specified using the probabilities
\( p(z_i=j) \) in E-step. Thus, in weighted D2-clustering algorithm, we set \( \rho_i = p(z_i=j) \) as the weight of signature \( l_i \).

After converging the prototype extraction algorithm, we calculate the average within cluster distances for \( C_j \) by the following equation:

\[
\bar{\pi}_j = \left( \sum_{i \in C_j} \rho_i u_{ij} \right) / \left( \sum_{i \in C_j} \rho_i \right)
\]

(19)

where \( u_{ij} \) is the Mallows distance between the signature \( l_i \) and the final prototype. If the average distance \( \bar{\pi}_j \) is greater than a predefined threshold, the cluster will be split into two clusters. To this end, a signature is randomly chosen from cluster \( C_j \) as a new prototype. Given the current prototype of cluster \( C_j \) and the new selected prototype, signatures in \( C_j \) are assigned to one of two prototypes based on their distances to the prototypes. After initializing each cluster with current two prototypes, the prototype extraction step is applied again to each cluster, separately. The splitting procedure is recursively applied to new clusters until the within cluster distance is less than a threshold. Note that the probability that the signature \( l_i \) is selected as a new prototype, is defined by the weight \( p_{\text{sel} i} = \rho_i / \sum_{k \in C_j} \rho_k \). Also, we use the value of splitting threshold as listed in Table 2.

Given the new prototypes, we update the parameters of Gamma distributions fitted into the clusters such that function \( Q(\theta) \) in Eq. (16) is maximized. In Appendix, we prove that the estimator of parameters \( \theta \) will be

\[
\hat{\theta}_j = \ln \left( \frac{\bar{\pi}_j}{\bar{s}} \right)
\]

(20)

Here \( \bar{\pi}_j \) is obtained using Eq. (19) and \( \psi(\cdot) \) is the di-Gamma function: \( \psi(s) = \log \Gamma(s)/ds \).

In addition to the above parameters, there is a prior probability \( o_j \) for the cluster \( C_j \) as defined in Eq. (12). To update \( o_j \), we set it to the sum of signatures' weights \( \rho_i \) in the cluster \( C_j \). Then, \( o_j \)'s are normalized to meet the constraint \( \sum_{j} o_j = 1 \) in all concepts.

4.3.3 Filtering irrelevant signatures

As stated above, the unlabeled images are shared between all concepts such that a large quantity of the training images for one concept comes from the unlabeled set. Since these images will be assigned to one cluster in every concept, we must deal with a large number of signatures in the update step. However, only a small number of these signatures are relevant to that cluster. The irrelevant signatures are divided into two categories: noisy ones and outlier. The noisy signatures do not belong to a concept but their color and texture is somehow similar to the signatures in that concept. They are mainly generated at feature extraction step. Outliers, on the other hand, are signatures that are generated by the density function of a cluster in another concept. They form most of the irrelevant signatures.

The density function of a cluster is defined based on Mallows distance between a signature and the prototype of cluster. Assume that signature \( l_i \) is an outlier for \( k \)th concept. Therefore, \( l_i \) will be far from all prototypes in \( k \)th concept. This issue leads to low values of \( p(z_i=j) \)'s for all clusters \( C_j \) in \( k \)th concept. Thus, we determine a subset of unlabeled signatures in the cluster \( C_j \) with values of \( p(z_i=j) \) less than a threshold. These signatures are discarded as outlier from the update process of cluster \( C_j \). Fig. 2 shows a typical concept with seven prototypes such that each cluster consists of some labeled and unlabeled signatures. Note that the size of each signature represents its weight. The signatures with low weights are considered as outlier and will be removed from the clusters.

To specify the filtering threshold, we use a stepwise decreasing function of the updating iterations. We start with an initial threshold at the first iteration of EM algorithm. The threshold for next iterations is obtained by multiplying the current threshold to a decreasing rate. The values of initial threshold and decreasing rate are listed in Table 2. The reason for using this strategy is that the early iterations of algorithm must use the most confident unlabeled images for updating the parameters of a cluster.

The filtering mechanism provides some advantages. At first, it reduces the computational time of EM algorithm because the number of constraints and variables \( w_{ij} \)'s in Eq. (6) will be substantially reduced. As stated in Ref. [11], the unlabeled samples will be informative for semi-supervised learning if the classes (concepts) are separated by a low density area. The information content of outliers is very low and they will increase the overlap between different concepts. The filtering process will reduce this negative effect of outliers and prevents EM algorithm from converging to local maxima.

4.4 Parameter estimation for initial Models

Since the initial models are constructed using the labeled images, the probabilities \( p(z_i=j) \)'s are equal to one for images in cluster \( C_j \). Therefore, we have the following equation for \( p(z_i=j) \):

\[
p(z_i=j) = \mathbb{I}(C_j = \text{Nearest}_k(l_i)) \times \mathbb{I}(\text{Concept}_k = k) \quad \forall k = 1, \ldots, T
\]

(21)

where \( T \) is the number of concepts. To compute the maximum likelihood estimation (MLE) of parameters for initial models, the above values of \( p(z_i=j) \) are used in Eq. (20).

Fig. 3 demonstrates the flowchart of training procedure with two main blocks: initial models construction and incorporating unlabeled images. Update process of one concept will be continued until the change in the log-likelihood function is less than a threshold. The value of this threshold is given in Table 2.
5. Experimental results

To evaluate the proposed annotation system, Corel60k dataset is used. Each test image is annotated in a similar manner as ALIPR [8]. Briefly, the joint probability of words and visual features are computed using the concepts probabilities. After sorting words based on their joint probabilities in descending order, the input image is annotated by the top ranked words.

To assess the annotation performance, we compute the average of precision and recall for different words as explained in [8]. In our experiments, we annotate all test images with different number of words (from 1 to 15) and compute precision and recall for each case study. We shall mention that all implementations are performed on MATLAB environment on a PC with 2.4 GHz Corei7 processor and 4 GB memory.

5.1. Parameters selection

Because the training step and parameters estimation of models are based on distance metric in Eq. (3), it is essential to properly specify the weights $\lambda_1$ and $\lambda_2$ in this metric. Using Spectral clustering in Section 4.1, we can obtain a group (cluster) of images with similar contents. To analyze the behavior of color and texture parts, we compute the Mallows distance among all pairs of color (texture) signatures of images in the same group. Fig. 4 shows the histograms of distance values obtained for color and texture parts. These histograms follow Gamma distribution with the parameters listed in Table 1. While the shape of two distributions is very similar, they have different scale parameters. According to Table 1, the ratio of scale parameter for texture distance to color distance is 0.42. Therefore, in Eq. (3), we set the color and texture weights as...


\[ \lambda_1 = 0.42 \text{ and } \lambda_2 = 1, \text{ respectively, to normalize the distances to the same scale before combining them.} \]

With the above weights, the average of distances between similar images will be 341.25. To consider the effect of noisy images in averaging, we set the splitting threshold of clusters in M-step to 300 which is slightly less than the average. In Table 2, we specify the values of parameters used in the modified EM algorithm. We also avoid splitting clusters with less than 5 images. In the filtering step, high values for initial threshold and decreasing rate ensure that the earliest iterations of update process incorporate most relevant images to a cluster.

Because prototype extraction procedure in Section 3 has a great influence on the accuracy of the generative models, we fix the threshold used in this procedure to a small value (i.e. \(10^{-5}\)). We consider ALIPR for comparison because it follows the same structure as our annotation system. The parameters of this approach are selected in the same strategy as discussed in above. However, we set \(\lambda_1 = 1\) and \(\lambda_2 = 1\) for ALIPR baseline as stated in Ref. [8].

5.2. Supervised performance

The first class of evaluation is performed under the supervised learning framework. To this end, we select the first 150 concepts from Corel60k and use the initial models of concepts for image annotation. Each concept contains 100 images and there are a total of 137 distinct words to describe the selected concepts. Histogram of the number of words assigned to the first 150 concepts of Corel60k dataset is shown in Fig. 5. This figure indicates that most images are annotated with 3 or 4 words. We use the first 80 images of each concept as the train set and the remaining images are used as test images. For a given test image, a word provided by the annotation system is correct if the corresponding concept of image contains that word.

In addition to ALIPR, we compared our results with Baseline annotation system [29]. This technique treats annotation as a retrieval problem and associates words to a new test image using its \(K\) nearest neighbors in the labeled set of images. The goal of this approach is to eliminate the need for training a complex model in the annotation system. It was shown [29] that this baseline method outperforms most of state of the art methods such as annotation system in [5]. To implement Baseline annotation system, we set \(K=5\) as suggested in Ref. [29] and compute the distance between images using Eq. (3). Other similar systems assume that there are annotations for each individual image and do not follow the PSU protocol. Implementation of these systems under PSU protocol does not provide informative comparison. Therefore, we do not consider them in our experiments.

Fig. 6 illustrates the results obtained by our approach under supervised framework against ALIPR and Baseline methods. As shown in this figure, our approach outperforms ALIPR in both recall and precision. As previously stated, the supervised setting differs from ALIPR in the feature extraction and initial clustering step. However, concept modeling is performed in a similar manner as ALIPR. This means that feature extraction and the initial clustering based on graph partitioning in our approach leads to more informative prototypes than those in ALIPR.

As shown in Fig. 6, Baseline system could not achieve good performance on Corel60k. Averaging over annotation metrics for 2–6 words, we conclude that our approach outperforms the Baseline system by 23% and 17% in precision and recall, respectively. The main reason for poor performance of the Baseline system is that images in Corel60k are annotated with noisy tags. This issue is discussed in more details in Section 5.4.

As discussed in Section 3, D2-clustering in ALIPR has a divisive nature which makes concept modeling computationally expensive.

\[ \text{Table 1} \]

ML estimation for parameters of distributions fitted into color and texture distances.

<table>
<thead>
<tr>
<th></th>
<th>Color distance</th>
<th>Texture distance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Shape (s)</strong></td>
<td>1.94</td>
<td>1.71</td>
</tr>
<tr>
<td><strong>Scale (b)</strong></td>
<td>217.76</td>
<td>91.65</td>
</tr>
</tbody>
</table>

\[ \text{Table 2} \]

Parameters used in the modified EM algorithm and their values.

<table>
<thead>
<tr>
<th>Parameter specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold for splitting clusters</td>
<td>300</td>
</tr>
<tr>
<td>Minimum number of images in a cluster</td>
<td>5</td>
</tr>
<tr>
<td>Initial value of filtering threshold</td>
<td>0.9</td>
</tr>
<tr>
<td>Decreasing rate of filtering threshold</td>
<td>0.9</td>
</tr>
<tr>
<td>Stopping threshold for EM algorithm</td>
<td>0.001</td>
</tr>
</tbody>
</table>

Fig. 5. Histogram of the number of words assigned to the first 150 concepts of Corel60k dataset.

Fig. 6. Comparative analysis of supervised setting with ALIPR [8] and Baseline system [29] with respect to precision (a) and recall (b). The horizontal axes show the number of words to annotate test images.
However, we avoid the divisive strategy by applying the spectral clustering on the image signatures. This will result in a great reduction in the required time for prototype extraction and subsequently the concept modeling. To experimentally study this benefit of our method, we examine the average time for constructing the model of one concept when there are 80 labeled images. The results are demonstrated in Table 3 which shows that the proposed method reduces the training time of ALIPR by 67%.

In our proposed system, it takes 7.39 s on average to annotate an image using the models of 150 concepts. This time is comparable to ALIPR. We utilize multi-thread scheme in Parallel Computing Toolbox of MATLAB to compute concepts probabilities. It also takes 2.77 s on average to extract the color and texture signatures for an image with size $384 \times 256$.

Fig. 7 shows the suggested words for some test images provided by our annotation system in the supervised setting. Also, the ground-truth annotations (human annotations) are shown for comparison. It should be noted that some words provided by the annotation system could be considered as correct annotation although they have not appeared in the ground-truth. For instance, the words “ocean” and “beach” provide a correct semantic for the left image in the first row of Fig. 7, but these words are not provided in the ground-truth annotation. These words are also considered as false annotations in evaluating recall and precision in Fig. 6.

### 5.3. Semi-supervised performance

In the semi-supervised annotation, running time complexity is due to iterations over both $E$-step process and updating the prototypes of clusters in $M$-step. Unfortunately, runtime of these two steps scales up significantly by increasing the number of unlabeled images. Moreover, we consider a subset of images in a concept as unlabeled samples. Thus, by increasing the number of concepts, more running time is required for training. On the other hand, the goal of semi-supervised annotation is to reach the same level of performance as supervised annotation by using much fewer labeled images. To reach this goal with an acceptable computational complexity, we reduced the number of concepts in SSL setting to the first 50 concepts of Corel60k. These concepts contain 5000 images, a de facto size for image annotation [29]. They are annotated with 88 distinct words.

To evaluate the annotation performance under SSL setting, we conduct 5-fold cross validation on images of every concept. In more details, we randomly split images of every concept into five

---

Table 3

Comparative analysis of the training time (s) in ALIPR and proposed method for extracting the model of one concept with 80 labeled images.

<table>
<thead>
<tr>
<th>System Annotation</th>
<th>ALIPR</th>
<th>The proposed supervised setting</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Average time</strong></td>
<td>476.93</td>
<td>153.91</td>
</tr>
<tr>
<td><strong>Standard deviation</strong></td>
<td>120.87</td>
<td>83.95</td>
</tr>
</tbody>
</table>

---

![Fig. 7. Comparing the annotations of the proposed method with human annotations for some test images.](image-url)
parts with equal size (i.e. 20 images) and select each of five parts as the test images. The remaining 80 images of every concept are incorporated into the training phase and are divided into labeled and unlabeled sets as follows. Among 80 images in the training set of one concept, the first \( l \) images are considered as the labeled samples for that concept. To study the annotation performance for different number of labeled images, \( l \) has been varied from 25 to 75 (\( l = 25, 35, 45, 55, 65, 75 \)). To form the unlabeled set, we use the remaining images of concepts which are not included in the test and labeled sets. This means that we select \( u \) images (\( u = 80 - l \)) from each concept as the unlabeled images. After computing precision and recall for each test part, we compute the average over all five parts to assess annotation performance.

With the above configuration, after convergence of SSL algorithm, the results in Fig. 8(a) and (b) are obtained. Fig. 8(c) and (d) also shows precision and recall values at the first iteration of SSL algorithm. As the algorithm starts from the constructed models on labeled images, Fig. 8(c) and (d) indicates the results of supervised setting for the first 50 concepts.

If we compute the differences of recall and precision at the first and last iterations of SSL algorithm, the amount of increases indicates how much the SSL algorithm could improve the annotation performance. Fig. 9 shows the average differences of recall and precision for different number of labeled images on 50 concepts of Corel60k. In this figure, the average value for a specific number of labeled images represents the mean of differences between the first and last iterations after annotating test images with different number of words (from 1 to 15). Referring to the results in Figs. 8 and 9, we conclude the following points:

- The values of precision and recall are improved by increasing the number of labeled images such that the highest accuracy is obtained for 65 and 75 labeled images. This indicates that the labeled images have more importance than unlabeled ones to construct accurate models for the concepts. This issue is not in conflict with the use of semi-supervised annotation whose aim is to reduce the dependency of modeling algorithm to the labeled images by incorporating unlabeled ones.

- According to the test configuration, there is an increase in the number of unlabeled images when the number of labeled images decreases. Fig. 9 reveals that the average differences of recall and precision values between the first and last iterations of SSL algorithm have increased when the number of unlabeled (labeled) images increases (decreases). As previously stated, the first iteration of semi-supervised algorithm is equivalent to the supervised version of the annotation system. Therefore, the amount of differences indicates the level of improvement provided by the unlabeled images. By...
comparing the results for different number of labeled images in Fig. 9, recall and precision are improved about 15% when there are 25 labeled images in every concept.

In addition to above experiments, we study changes in precision and recall when the number of concepts varies. In this line, we conduct the SSL algorithm on 30 and 40 concepts which contain 61 and 75 words in their descriptions, respectively. The results are shown in Figs. 10 and 11. By comparing the results in Figs. 8, 10 and 11, we conclude that the precision and recall values have decreased by increasing the number of concepts in the modeling process. To study this issue in more details, we compute the average of precision and recall values in each experiment (i.e. 30, 40 and 50 concepts) by varying the number of annotation words from 2 to 6. The results are shown in Fig. 12. The main reason for considering 2–6 words is that most images are annotated with this number of words as shown in Fig. 5.

Referring to Fig. 12, it is obvious that both precision and recall have decreased by increasing the number of concepts. The main reason is that increasing the number of concepts leads to more overlap between concepts and more increase in the misclassification probability as well. By decreasing the number of concepts from 50 to 40 and 30, on average, we achieve increasing rates of 16.98% and 33.67% in precision, respectively. For recall, these rates are 19.11% and 38.25%. This indicates that recall is more affected than precision when the number of concepts decreases.

5.4. Comparing with other approaches

We compare the proposed annotation system with iterative bi-relational graph-based (I-BG) method [15], a graph-based semi-supervised annotation. This approach constructs two subgraphs for images and semantic labels. It provides some links between them to obtain an integrated graph for annotation. To construct the subgraph

---

Fig. 10. Precision-recall curves of semi-supervised annotation on 30 concepts from Corel60k dataset.

Fig. 11. Precision-recall curves of semi-supervised annotation on 40 concepts from Corel60k dataset.

Fig. 12. The average of precision and recall values for different number of labeled images when the number of concepts is 30, 40 and 50.
of semantic labels, we follow same strategy as the I-BG method. After computing the distance between every pair of images using Eq. (3), we follow the same strategy as in Ref. [15] to form the image subgraph. This provides a fair comparison between our approach and I-BG because the same distance metric is used in both methods. Furthermore, we set $\alpha=0.01$ and $\beta=0.05$ for I-BG as suggested in Ref. [15].

In addition to I-BG, we compare our semi-supervised annotation with Baseline system [29] introduced in Section 5.2. Similar to the test configuration in Section 5.3, we assess the performance of I-BG and Baseline systems with different numbers of labeled images. In I-BG method, all images are incorporated into graph construction but a subset of images is considered as the labeled set. On the other hand, the Baseline method only utilizes the labeled set for annotating test images.

We apply I-BG and Baseline methods to 50 concepts of Corel60k with 5-fold cross validation. On average, each image in Corel60k contains 4.1 words. Therefore, we evaluate the performance of three methods after annotating all test images with 4 words. As shown in Fig. 13, our approach outperforms both I-BG and Baseline methods. Computing total average of precision values for different number of images, we conclude that there is an increase of 6% and 23% in the proposed approach with respect to I-BG and Baseline methods, respectively. This increase is about 9% and 30% for recall values. Note that Baseline has a poor performance on Corel60k dataset. As discussed in the previous section, noisy annotations and high intra-concepts variance in Corel60k will highly degrade the performance of Baseline system that extracts the tags of a test image using its top K nearest neighbors in the labeled set.

Finally, it takes about 33.23 s (on average) to annotate an image using the I-BG method. This time is about 5.18 s for our approach. Thus, we can conclude that running time of graph-based methods is highly more than the generative models as discussed in Section 1.1.

### 5.5. Applying our approach to image-level annotated datasets

In this section, we discuss how our approach is applied to a dataset whose tags are provided at image-level (i.e. tags are assigned to each individual image instead of image concept). As a case study, we consider IAPR TC-12 dataset [30] containing 20,000 images. This dataset is split into 16,000 training images and 4000 test images. There are 228 distinct words that appear in both training and test sets.

As mentioned before, we assume that images are annotated at concept-level. The main reason for this assumption is to guarantee that each semantic class has enough images (samples) in the training phase. Indeed, the parameters of Gamma distribution fitted into a cluster can be accurately estimated if it contains enough samples. Therefore, to apply our approach to an image-level annotated dataset, we must provide semantic classes with enough training images. Thus, we consider each word of dataset as a semantic class and exclude those words from the training phase which were assigned to a number of images less than a predefined threshold. Table 4 shows the number of distinct words in IAPR dataset which are assigned to more than 20, 50 and 100 images. For instance, the first row of Table 4 indicates that there are 163 words that appear in more than 20 images. Moreover, there are 15970 images in the dataset which are annotated by 163 words. In below, we discuss the results on IAPR with threshold value of 50 and 100 for the number of images.

At first, we assess the performance of SSL algorithm on IAPR dataset when the value of threshold is 100. To compute the annotation metrics (i.e. precision and recall), we annotate each test image using the top 5 words with the largest concept probabilities. Moreover, to form labeled and unlabeled images in the training phase of SSL algorithm, we randomly select $l$ percent ($l=50, 60, 70, 80, 90$) of images assigned to each word as labeled images and the remaining as unlabeled samples. Fig. 14 shows precision and recall values in the first and final iterations of SSL algorithm for different values of $l$. Comparing the results in the first iteration (using only the labeled images) and the final iteration (after incorporating unlabeled images), we conclude that SSL algorithm improves the annotation performance significantly. As shown in Fig. 14, decreasing the percentage of labeled images which is equivalent to increasing the number of unlabeled images, leads to more difference between the first and final iterations in both precision and recall. However, as the percentage of labeled images increases, precision and recall values approach to the values obtained for 100% labeled images.

We also analyze the performance of SSL algorithm when the threshold for the number of images assigned to each word is set to 50. The results are shown in Fig. 15 indicating considerable increase in annotation metrics after convergence (i.e. the final iteration of SSL algorithm). In Fig. 16, we compare the results in Figs. 14 and 15 by computing the increasing rate of annotation metrics when the threshold changes from 50 to 100. Fig. 16 indicates that decreasing the percentage of labeled images leads to a rise the increasing rate of annotation metrics. When the threshold value changes from 100 to 50, some new semantic classes (words), each corresponding to a few images are incorporated in the training phase. As a result, the generalization of mixture models constructed for the new semantic classes will decrease as well. Another point concluded from Fig. 16 is that recall increases more than precision after discarding new semantic classes.

![Fig. 13. Comparing our approach with I-BG and Baseline methods with respect to (a) precision and (b) recall.](image)
6. Conclusions

Image annotation plays a major role for indexing the images in large datasets and photo-sharing communities. We proposed an annotation system in the semi-supervised framework to incorporate unlabeled images into the training phase. To this end, we proposed a generative modeling approach in two steps. In the first step, an initial mixture model is constructed for each concept using its labeled images. This step is equivalent to the maximum likelihood (ML) estimation of parameters in mixture models based on the labeled images. In the second step, the parameters of mixture models are updated through incorporating the unlabeled images. Since the unlabeled images generate incomplete observations for clusters, we customized EM algorithm steps to use the unlabeled signatures for updating parameters of Gamma distributions fitted into the clusters. In spite of many semi-supervised learning algorithms, the proposed approach is inductive and could annotate every test image not included in the training phase. Comparing to the supervised annotation systems, it reaches higher accuracy after incorporating the unlabeled samples. However, in our algorithm, the training phase needs more time than the supervised annotation, because we used iterative EM algorithm with an iterative nature. Our experiments also reveal that SSL algorithm will improve the annotation metrics for the datasets which are not organized based on PSU protocol.

There are many directions for the future works. One of the main challenges is related to the extraction of proper features from images. Using multiple descriptors for texture or new features such as shape could improve the annotation results. Our system does not consider the relation between different words. A suitable model for words correlations or their co-occurrences could help to refine unrelated words assigned to an image.

Conflict of interest statement

None declared.

Acknowledgements

We would like to thank Iran National Science Foundation (INSF) for their partial support of this work.

Appendix

In this section, we prove the parameter estimation of Gamma distribution in Eq. (20). Given the probabilities \( p(z_i=j) \) in the E-step, we must maximize function \( Q(\theta) \) defined in Eq. (16) in the

---

Fig. 14. Precision and recall values on IAPR dataset in the first and final iterations of SSL algorithm after discarding the words which are assigned to less than 100 images. The values of X-axis indicate the percentage of labeled images which are randomly selected from training images assigned to each word.

Fig. 15. Annotation performance of SSL algorithm on IAPR dataset in the presence of different percentages of labeled images after discarding words assigned to less than 50 images.

Fig. 16. Increasing rate of annotation metrics when the threshold for discarding the words of dataset changes from 50 to 100.
M-step. Substituting the definition of \( f(u_i|\theta) \) in Eq. (10) to Eq. (6), we have

\[
Q(\theta) = \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \log u_i - s \log \frac{b_j}{b_j + b_0} - \log \Gamma(s)
\]

(22)

If the parameter \( s \) is assumed to be fixed, term \( s \) will be a concave function of \( b_j \). After calculating the first derivative of \( s \) with respect to \( b_j \) and setting it to zero, we have

\[
\frac{\partial Q(\theta)}{\partial b_j} = \sum_{i=1}^{N} p(z_i=j) \left( \frac{s}{b_j} - \frac{u_i}{b_j} \right) = 0
\]

\[
\Rightarrow \frac{s}{b_j} = \frac{1}{b_j} \sum_{i=1}^{N} p(z_i=j) u_i
\]

(23)

To estimate \( b_j \), we obtain the following equation:

\[
b_j = \frac{\sum_{i=1}^{N} p(z_i=j) u_i}{\sum_{i=1}^{N} p(z_i=j)} \Rightarrow b_j = \frac{\bar{u}_j}{s}
\]

(24)

where \( \bar{u}_j \) is defined in Eq. (19). By substituting the above equation to Eq. (6), it will be

\[
Q(\theta) = \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \left( (s-1) \log u_i - s \log \left( \frac{\bar{u}_j}{s} \right) - \log \left( \frac{b_j}{\bar{b}_j} \right) - \log b_0 \right)
\]

\[
= \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \left[ (s-1) \log \left( \frac{u_i}{\bar{u}_j} \right) - \log u_i + s \log \frac{b_j}{\bar{b}_j} \right] - \log b_0
\]

(25)

Here, \( Q(\theta) \) is a concave function of \( s \) because \( \Gamma(s) \) is a convex function. Computing the derivative of \( Q(\theta) \) with respect to \( s \) and setting it to zero, we have

\[
\frac{\partial Q(\theta)}{\partial s} = \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \left[ (s-1) \log \left( \frac{u_i}{\bar{u}_j} \right) + \log s + 1 - \psi(s) \right]
\]

\[
= (\log s - \psi(s)) \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) + \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \left( \frac{u_i}{\bar{u}_j} \right)
\]

\[+ \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \log \frac{u_i}{\bar{u}_j}
\]

(26)

Note that term \( A \) is equal to zero

\[
\sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) - \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) = \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) - \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) = 0
\]

\[
- \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) = \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) = \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) = 0
\]

(27)

Therefore, Eq. (26) will be

\[
\frac{\partial Q(\theta)}{\partial s} = (\log s - \psi(s)) \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) + \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \log \frac{u_i}{\bar{u}_j}
\]

(28)

Setting above equation to zero, the estimation of \( s \) is solved by the following equation:

\[
\log s - \psi(s) = \left[ - \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \log \frac{u_i}{\bar{u}_j} \right] / \left[ \sum_{j=1}^{M} \sum_{i=1}^{N} p(z_i=j) \right]
\]

(29)
Mansour Jamzad has obtained his M.S. degree in computer science from McGill University, Montreal, Canada and his Ph.D in electrical engineering from Waseda University, Tokyo, Japan in 1989. For a period of two years after graduation he worked as a Post Doctorate Researcher in the Department of Electronics and Communication Engineering, Waseda University. He was a visiting researcher at Tokyo Metropolitan Institute of Gerontology during 1990–1994. He became a faculty member at Department of Computer Engineering, Sharif University of Technology, Tehran, Iran in 1995. Since then, he has been teaching digital image processing and machine vision graduate courses. His main research interests are related to digital image processing, machine vision and their applications. In recent years he has been working on watermarking, steganography, content-based image retrieval, image annotation, face gestures, robot vision and computer graphics.