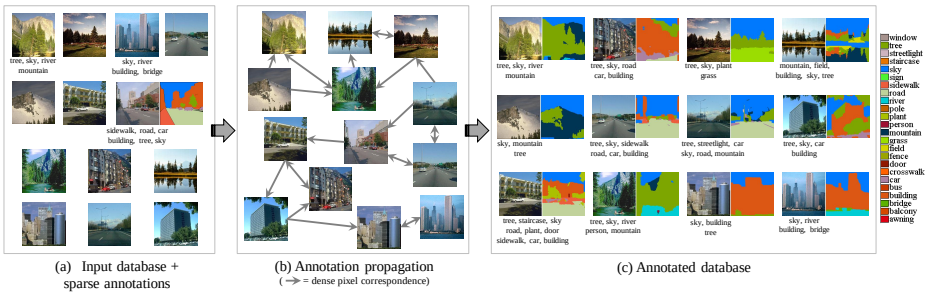
Annotation Propagation in Large Image Databases via Dense Image Correspondence

Michael Rubinstein[1,2], Ce Liu[2], and William T. Freeman[1]

[1]MIT CSAIL [2]Microsoft Research New England

Abstract. Our goal is to automatically annotate many images with a set of word tags and a pixel-wise map showing where each word tag occurs. Most previous approaches rely on a corpus of training images where each pixel is labeled. However, for large image databases, pixel labels are expensive to obtain and are often unavailable. Furthermore, when classifying multiple images, each image is typically solved for independently, which often results in inconsistent annotations across similar images. In this work, we incorporate dense image correspondence into the annotation model, allowing us to make do with significantly less labeled data and to resolve ambiguities by propagating inferred annotations from images with strong local visual evidence to images with weaker local evidence. We establish a large graphical model spanning all labeled and unlabeled images, then solve it to infer annotations, enforcing consistent annotations over similar visual patterns. Our model is optimized by efficient belief propagation algorithms embedded in an expectation-maximization (EM) scheme. Extensive experiments are conducted to evaluate the performance on several standard large-scale image datasets, showing that the proposed framework outperforms state-of-the-art methods.



(a) Input database + sparse annotations (b) Annotation propagation (→ = dense pixel correspondence) (c) Annotated database

Fig. 1. Input and output of our system. (a) The input image database, with scarce image tags and scarcer pixel labels. (b) Annotation propagation over the image graph, connecting similar images and corresponding pixels. (c) The fully annotated database by our system (more results can be found in Fig. 5 and the supplemental material).

1 Introduction

We seek to annotate a large set of images, starting from a partially annotated set. The resulting annotation will consist of a set of word tags for each image, as well as per-pixel labeling indicating where in the image each word tag appears. Such automatic annotation will be very useful for Internet image search, but will also have applications in other areas such as robotics, surveillance, and public safety.

In computer vision, scholars have investigated image annotation in two directions. One methodology uses *image similarities* to transfer textual annotation from labeled images to unlabeled samples, under the assumption that *similar images should have similar annotations*. Notably, Makadia *et al*. [1] recently proposed a simple baseline approach for auto-annotation based on global image features, and a greedy algorithm for transferring tags from similar images. ARISTA [2] automatically annotated a web dataset of billions of images by transferring tags via near-duplicates. Although those methods have clearly demonstrated the merits of using similar images to transfer annotations, they do so only between globally very similar images, and cannot account for locally similar patterns.

Consequently, the other methodology focused on dense annotation of images, known as *semantic labeling* [3–6], where correspondences between text and local image features are established for annotation propagation: *similar local features should have similar labels.* These methods often aim to label each pixel in an image using models learned from a training database. In [7] and [8], relationships between text and visual words are characterized by conventional language translation models. More recently, Shotton *et al*. [3] proposed to train a discriminative model based on the texton representation, and to use conditional random fields (CRF) to combine various cues to generate spatially smooth labeling. This approach was successively extended in [4], where efficient randomized decision forests are used for significant speedup. Liu *et al*. [5] proposed a nonparametric approach to semantic labeling, where text labels are transferred from a labeled database to parse a query image via dense scene correspondences.

Since predicting annotation from image features is by nature ambiguous (*e.g*. textureless regions can be *sky*, *wall*, or *ceiling*), such methods rely on regularities in a large corpus of training data of pixel-wise densely labeled images. However, for large image databases, high-quality pixel labels are very expensive to obtain. Furthermore, the annotation is typically computed *independently* for each image, which often results in inconsistent annotations due to visual ambiguities in image-to-text.

In this work, we show that we can overcome these drawbacks by incorporating *dense image correspondences* into the annotation model. Image correspondences can help alleviate local ambiguities by propagating annotations in images with strong local evidence to images with weaker evidence. They also implicitly add consideration of contextual information.

We present a principled framework for automatic image annotation based on dense labeling and image correspondence. Our model is guided by four assumptions: (a) regions corresponding to each tag have distinct visual appearances, (b) neighboring pixels in one image tend to have the same annotation, (c) similar patterns across different images should have similar annotation, and (d) tags and tag co-occurrences which are more frequent in the database are also more probable.

By means of dense image correspondence [9], we establish a large image graph across all labeled and unlabeled images for consistent annotations over similar image patterns across different images, where annotation propagation is solved *jointly* for the entire database rather than for a single image. Our model is effectively optimized by efficient belief propagation algorithms embedded within an expectation-maximization (EM) scheme, alternating between visual appearance modeling (via Gaussian mixture

models) and pixel-wise tag inference. We conducted extensive experiments on standard large-scale datasets, namely LabelMe [10], ESP [11] and IAPR [12], showing that our system consistently outperforms the state-of-the-art in automatic annotation and semantic labeling, while requiring significantly less labeled data.

In summary, this paper makes the following contributions. (a) We show that dense correspondences between similar images can assist in resolving text-to-image visual ambiguities, and can dramatically decrease the amount of training data for automatic annotation. (b) We propose a novel graphical model for solving image annotation *jointly* by building an dense image graph that spans the entire database for propagating annotations. (c) Our novel formulation assumes scarce data, and combines different levels of supervision - tagged images (cheaper to obtain) and labeled images (expensive to obtain). We thoroughly studied how the ratio of tags and labels affects the performance.

2 Terminology and Formulation

Textual tagging and pixel-level labeling are both plausible types of "annotations" for an image. In this paper, we will consistently refer to semantic, pixel-level labeling (or semantic segmentation) as "labeling" and to textual image-level annotation as "tags". The set of tags of an image is comprised of "words" from a vocabulary. These words can be specified by a user, or obtained from text surrounding an image on a webpage.

The input database $\mathbf{\Omega} = (\mathbf{I}, \mathbf{V})$ is comprised of RGB images $\mathbf{I} = \{I_1, \ldots, I_N\}$ and vocabulary $\mathbf{V} = \{l_1, \ldots, l_L\}$. We treat auto-annotation as a labeling problem, where the goal is to compute the labeling $\mathbf{C} = \{c_1, \ldots, c_N\}$ for all images in the database, where for pixel $\mathbf{p} = (x, y)$, $c_i(\mathbf{p}) \in \{1, \ldots, L, \varnothing\}$ ($\varnothing$ denoting *unlabeled*) indexes into vocabulary $\mathbf{V}$. The tags associated with the images, $\mathbf{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_N : \mathbf{t}_i \subseteq \{1, ..., L\}\}$, are then defined directly as the set union of the pixel labels $\mathbf{t}_i = \cup_{\mathbf{p} \in \Lambda_i} c_i(\mathbf{p})$, where Λ_i is image I_i's lattice.

Initially, we may be given annotations for some images in the database. We denote by $\mathbf{I}_t$ and $\mathbf{T}_t$, and $\mathbf{I}_l$ and $\mathbf{C}_l$ the corresponding subsets of *tagged* images with their tags, and *labeled* images with their labels, respectively. Also let (r_t, r_l) be the ratio of the database images that are tagged and labeled, respectively.

We formulate this discrete labeling problem in a probabilistic framework, where our goal is to construct the joint posterior distribution of pixel labels given the input database and available annotations, $P(\mathbf{C}|\mathbf{\Omega}, \mathbf{T}_t, \mathbf{C}_l)$. We assume that (a) corresponding patterns across images should have similar annotation, and (b) neighboring pixels in one image tend to have the same annotation. This effectively factorizes the distribution as follows. Written in energy form ($-\log P(\mathbf{C}|\mathbf{\Omega}, \mathbf{T}_t, \mathbf{C}_l)$), the cost of global assignment $\mathbf{C}$ of labels to pixels is given by

$$E(\mathbf{C}) = \sum_{i=1}^{N} \sum_{\mathbf{p} \in \Lambda_i} \left[\Phi_p(c_i(\mathbf{p})) + \sum_{\mathbf{q} \in \mathcal{N}_\mathbf{p}} \Psi_{int}\left(c_i(\mathbf{p}), c_i(\mathbf{q})\right) + \sum_{j \in \mathcal{N}_i} \Psi_{ext}\left(c_i(\mathbf{p}), c_j(\mathbf{p} + \mathbf{w}_{ij}(\mathbf{p}))\right) \right], \tag{1}$$

where $\mathcal{N}_\mathbf{p}$ are the spatial neighbors of pixel $\mathbf{p}$ within its image, and $\mathcal{N}_i$ are other images similar to image I_i. For each such image $j \in N_i$, $\mathbf{w}_{ij}$ defines the (dense) correspondence between pixels in I_i and pixels in I_j.

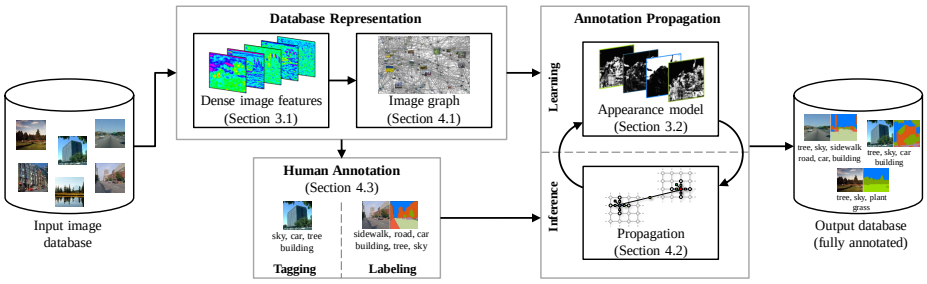


Fig. 2. System overview. Each rectangle corresponds to a module in the system.

This objective function defines a (directed) graphical model over the entire image database. We define the data term, Φ_p, by means of local visual properties of image I_i at pixel $\mathbf{p}$. We extract features from all images and learn a model to correspond vocabulary words with image pixels. Since visual appearance is often ambiguous at the pixel or region level, we use two types of regularization. The *intra-image* smoothness term, ψ_{int}, is used to regularize the labeling with respect to the image structures, while the *inter-image* smoothness term, ψ_{ext}, is used for regularization across corresponding pixels in similar images.

Fig. 2 shows an overview of our system, and Fig. 4 visualizes this graphical model. In the upcoming sections we describe in detail how we formulate each term of the objective function, how we optimize it *jointly* for all pixels in the database, and show experimental results of this approach.

3 Text-to-Image Correspondence

To characterize text-to-image correspondences, we first estimate the probability distribution $P_a(c_i(\mathbf{p}))$ of words occurring at each pixel, based on local visual properties. For example, an image might be tagged with *car* and *road*, but their locations within the image are unknown. We leverage the large number of images and the available tags to correlate the database vocabulary with image pixels. We first extract local image features for every pixel, and then learn a visual appearance model for each word by utilizing visual commonalities among images with similar tags, and pixel labels (if available).

3.1 Local Image Descriptors

We selected features used prevalently in object and scene recognition to characterize local image structures and color features. Structures are represented using both SIFT [9] and HOG [13] features. We compute dense SIFT descriptors with 3 and 7 cells around a pixel to account for scales. We then compute HOG features and stack together neighboring HOG descriptors within 2×2 patches [14]. Color is represented using a 7×7 patch in L*a*b color space centered at each pixel. Stacking all the features yields a 527-dimensional descriptor $D_i(\mathbf{p})$ for every pixel $\mathbf{p}$ in image I_i. We use PCA to reduce the descriptor to $d = 50$ dimensions, capturing approximately 80% of the features' variance.

3.2 Learning Appearance Models

We use a generative model based on Gaussian mixtures to represent the distribution of the above continuous features. More specifically, we model each word in the database vocabulary using a full-covariance Gaussian Mixture Model (GMM) in the 50D descriptor space. Such models have been successfully applied in the past to model object appearance for image segmentation [15]. Note that our system is not limited to work with this particular model. In fact, we also experimented with a discriminative approach, Randomized Forests [16], previously used for semantic segmentation [4] as an alternative to GMM. We found that GMM produces better results than random forests in our system (see Section 5).

For pixel $\mathbf{p}$ in image I_i, we define

$$P(D_i(\mathbf{p});\boldsymbol{\Theta}) = \sum_{l=1}^{L}\left(\rho_l \sum_{k=1}^{M} \pi_{l,k}\mathcal{N}\left(D_i(\mathbf{p});\boldsymbol{\mu}_{l,k},\boldsymbol{\Sigma}_{l,k}\right)\right) + \rho_\epsilon \mathcal{N}\left(D_i(\mathbf{p});\boldsymbol{\mu}_\epsilon,\boldsymbol{\Sigma}_\epsilon\right), \quad (2)$$

where ρ_l is the weight of model (word) l in generating the feature $D_i(\mathbf{p})$, M is the number of components in each model ($M = 5$), and $\boldsymbol{\theta}_l = \left(\pi_{l,k}, \boldsymbol{\mu}_{l,k}, \boldsymbol{\Sigma}_{l,k}\right)$ is the mixture weight, mean and covariance of component k in model l, respectively. We use a Gaussian outlier model with parameters $\boldsymbol{\theta}_\epsilon = (\boldsymbol{\mu}_\epsilon, \boldsymbol{\Sigma}_\epsilon)$ and weight ρ_ϵ. The intuition for the outlier model is to add an *unlabeled* word to the vocabulary $\mathbf{V}$. $\boldsymbol{\Theta} = (\{\rho_l\}_{l=1:L}, \rho_\epsilon, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_L, \boldsymbol{\theta}_\epsilon)$ is a vector containing all parameters of the model.

We optimize for $\boldsymbol{\Theta}$ in the maximum likelihood sense using a standard EM algorithm. We initialize the models by partitioning the descriptors into L clusters using k-means and fitting a GMM to each cluster. The outlier model is initialized from randomly selected pixels throughout the database. We also explicitly restrict each pixel to contribute its data to models of words corresponding to its estimated (or given) image tags only. That is, we clamp the posteriors to zero for all $l \notin \mathbf{t}_i$. For labeled images $\mathbf{I}_l$, we keep the posteriors fixed according to the given labels (setting zero probability to all other labels). To account for partial annotations, we introduce an additional weight α_i for all descriptors of image I_i, set to α_t, α_l or 1 (we use $\alpha_t = 5, \alpha_l = 10$) according to whether image I_i was tagged, labeled, or inferred automatically by the algorithm, respectively. More details can be found in the supplementary material. Given the learned model parameters, $\boldsymbol{\Theta}$, and an observed descriptor, $D_i(\mathbf{p})$, the probability of the pixel belonging to word l is computed by $P_a(c_i(\mathbf{p}) = l; D_i(\mathbf{p}), \boldsymbol{\Theta}) = \frac{\rho_l \sum_{k=1}^{M} \pi_{l,k}\mathcal{N}\left(D_i(\mathbf{p});\boldsymbol{\mu}_{l,k},\boldsymbol{\Sigma}_{l,k}\right)}{\sum_{j=1}^{L}\left(\rho_j \sum_{k=1}^{M} \pi_{j,k}\mathcal{N}\left(D_i(\mathbf{p});\boldsymbol{\mu}_{j,k},\boldsymbol{\Sigma}_{j,k}\right)\right)+\rho_\epsilon \mathcal{N}(D_i(\mathbf{p});\boldsymbol{\mu}_\epsilon,\boldsymbol{\Sigma}_\epsilon)}$.

Fig. 3 (b) and (c) show an example result of the algorithm on an image from LabelMe Outdoors dataset (See Section 5 for details on the experiment). Clearly, our model is able to capture meaningful correlations between words and image regions.

4 Annotation Propagation

Now that we have obtained reasonable local evidences, we can utilize image information and correspondences between images to regularize the estimation and propagate annotations between images in the database. To this end, we compute dense pixel-wise correspondences and construct an image graph connecting similar images and corresponding pixels.

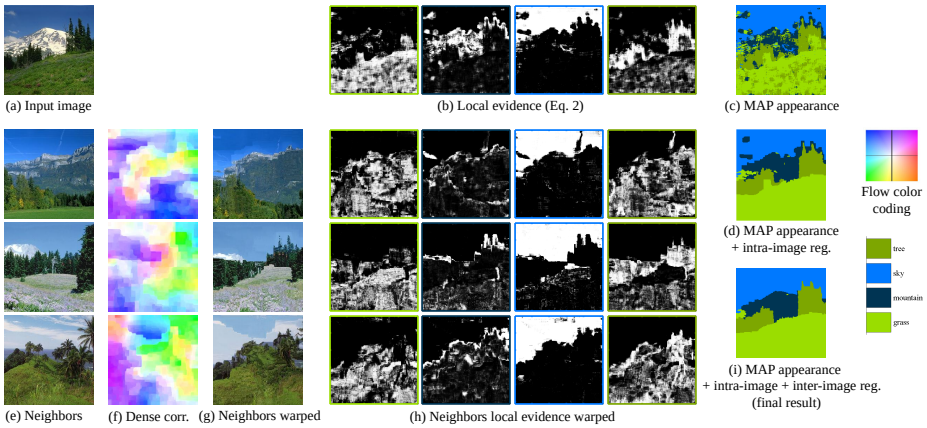


Fig. 3. Text-to-image and dense image correspondences. (a) An image from LabelMe Outdoors dataset. (b) Visualization of the local evidence P_a for the four most probable words, colored from black (low probability) to white (high probability). (c) The MAP classification based on local evidence alone, computed *independently* at each pixel. (d) The classification with spatial regularization (Eq. 8). (e-g) Nearest neighbors of the image in (a) and dense pixel correspondences with (a). (h) Same as (b) for each of the neighbors, warped towards the image (a). (i) The final MAP labeling using inter-image regularization (Eq. 7).

4.1 Dense Image Correspondence

We first fetch the top $\langle K, \epsilon \rangle$ nearest neighbors, $\mathcal{N}_i$, for every image I_i using the GIST descriptor [17], where K is the maximum number of neighbors, and ϵ is a threshold on the distance between the images (above which neighbors are discarded) [5]. We use SIFT-flow [9] to align the image with each of its neighbors, which results in an integer warp field $\mathbf{w}_{ij}$ mapping each pixel in I_i to a pixel in I_j. Notice that neither the nearest neighbors nor $\mathbf{w}_{ij}$ need necessarily be symmetric (as indicated by the arrows in Fig. 1(b)).

4.2 Large-scale Inference

Data term. The data term is comprised of four components:

$$\Phi_p\Big(c_i(\mathbf{p}) = l\Big) = -\log P_a(c_i(\mathbf{p})) - \log P_t^i(l) - \lambda_s \log P_s(c_i(\mathbf{p})) - \lambda_c \log P_c^i(c_i(\mathbf{p})), \tag{3}$$

where $P_a(c_i(\mathbf{p}) = l; D_i(\mathbf{p}), \mathbf{\Theta})$ was defined in the previous section, $P_t^i(l)$ is the estimated probability of image I_i having the tag l, and $P_s(c_i(\mathbf{p}))$ and $P_c^i(c_i(\mathbf{p}))$ capture the probability of the pixel $\mathbf{p}$ having the word l based on its relative spatial position and color, respectively. We use superscript i in P_t^i and P_c^i to emphasize that they are estimated separately for each image. λ_s, λ_c balance the contribution of P_s and P_c^i, respectively.

The term $P_t^i(l)$ is used to bias the tags of image I_i towards ones with higher frequency and co-occurrence among its neighbors. We estimate this probability as

$$P_t^i(l) = \frac{\beta}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \delta\left[l \in \mathbf{t}_j\right] + \frac{1-\beta}{Z} \sum_{j \in \mathcal{N}_i} \sum_{m \in \mathbf{t}_j} \mathbf{h}^o(l, m), \tag{4}$$

where $\delta[\cdot]$ evaluates to 1 when the inside condition is true, $\mathbf{h}^o$ is the $L \times L$ row-normalized tag co-occurrence matrix, computed from the current tag estimates and initialized from the known tags, and $Z = \sum_{j \in \mathcal{N}_i} |\mathbf{t}_j|$. The first term in Eq. 4 measures the frequency of word l among image I_i's neighbors, and the second term is the mean co-occurrence rate of word l within its neighbors' tags. We typically set $\beta = 0.5$, assigning equal contribution to the two terms. This term is inspired by [1], but we do not set explicit threshold on the number of tags to infer as they do.

Both the spatial and color terms are computed from the current pixel label estimates. The spatial location term is computed as

$$P_s\Big(c_i(\mathbf{p}) = l\Big) = \mathbf{h}_l^s(\mathbf{p}), \tag{5}$$

where $\mathbf{h}_l^s(\mathbf{p})$ is the normalized spatial histogram of word l across all images in the database. This term will assist in places where the appearance and pixel correspondence might not be as reliable (see Fig. 2 in the supplemental material).

The color term will assist in refining the labels internally within the image, and is computed as

$$P_c^i\Big(c_i(\mathbf{p}) = l\Big) = \mathbf{h}_c^{i,l}(I_i(\mathbf{p})), \tag{6}$$

where $\mathbf{h}_c^{i,l}$ is the color histogram of word l in image I_i. We use 3D histograms of 64 bins in each of the color channels to represent $\mathbf{h}_c^{i,l}$.

Smoothness terms. To encourage corresponding pixels in similar images to attain the same word, we define the *inter-image* compatibility between corresponding pixels $\mathbf{p}$ and $\mathbf{r} = \mathbf{p} + \mathbf{w}_{ij}(\mathbf{p})$ in images I_i and I_j respectively as

$$\Psi_{ext}\Big(c_i(\mathbf{p}) = l_p, c_j(\mathbf{r}) = l_r\Big) = \delta\left[l_p \neq l_r\right] \frac{\alpha_j}{\alpha_i} \lambda_{ext} \exp\Big(-|S_i(\mathbf{p}) - S_j(\mathbf{r})|\Big), \tag{7}$$

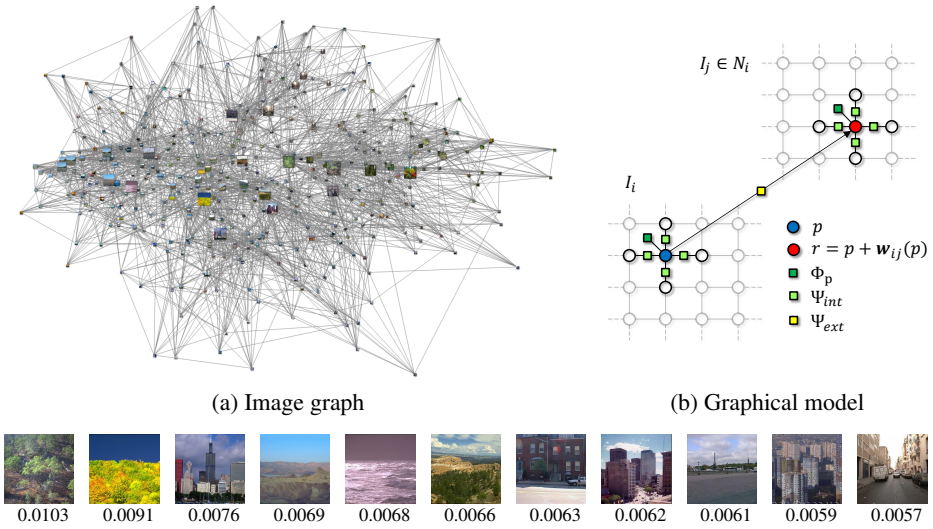
where α_i, α_j are the image weights as defined in Section 3.2, and S_i is the SIFT descriptor for image I_i. Intuitively, better matching between corresponding pixels will result in higher penalty when assigning them different labels, weighted by the relative importance of the neighbor's label.

We define the *intra-image* compatibility between neighboring pixels based on tag co-occurrence and image structures. For image I_i and spatial neighbors $\mathbf{p}, \mathbf{q} \in \mathcal{N}_\mathbf{p}$,

$$\Psi_{int}\Big(c_i(\mathbf{p}) = l_p, c_i(\mathbf{q}) = l_q\Big) = -\lambda_o \log \mathbf{h}^o\left(l_p, l_q\right) + \delta\left[l_p \neq l_q\right] \lambda_{int} \exp\Big(-\|I_i(\mathbf{p}) - I_i(\mathbf{q})\|\Big). \tag{8}$$

Overall, the free parameters of this graphical model are $\{\lambda_s, \lambda_c, \lambda_o, \lambda_{int}, \lambda_{ext}\}$, which we tune manually. We believe these parameters could be learned from a database with ground-truth labeling, but leave this for future work. .

Inference. We use a parallel coordinate descent algorithm to optimize Eq. 1, which alternates between estimating the appearance model and propagating annotations between pixels. The appearance model is initialized from the images and partial annotations in the database. Then, we partition the message passing scheme into intra- and inter-image updates, parallelized by distributing the computation of each image to a different core. Message passing is performed using efficient parallel belief propagation, while intra-image update iterates between belief propagation and re-estimating the color models (Eq. 6) in a GrabCut fashion [18]. Once the algorithm converges, we compute the MAP labeling that determines both labels and tags for all the images.



(a) Image graph (b) Graphical model

(c) Image hubs used for human labeling and tagging

Fig. 4. The image graph serves as an underlying representation for inference and human labeling. (a) Visualization of the image graph of the LMO dataset using 400 sampled images and $K = 10$. The size of each image corresponds to its visual pagerank score. (b) The graphical model. Each pixel in the database is connected to spatially adjacent pixels in its image and corresponding pixels in similar images. (c) Top ranked images selected automatically for human annotation. Underneath each image is its visual pagerank score.

4.3 Choosing Images to Annotate

As there is freedom to choose images to be labeled by the user, intuitively, we would want to strategically choose "image hubs" that have many similar images, as they will have many direct neighbors in the image graph to which they can propagate labels efficiently. We use visual pagerank [19] to find good images to label, using the GIST descriptor as the image similarity measure. To make sure that images throughout the database are considered, we initially cluster the images and use a non-uniform damping factor in the visual rank computation (Eq. 2 in [19]), assigning higher weight to the images closest to the cluster centers. Given an annotation *budget* (r_t, r_l), we then set $\mathbf{I}_l$ and $\mathbf{I}_t$ as the r_l and r_t top ranked images, respectively. Fig. 4(c) shows the top image hubs selected automatically with this approach, which nicely span the variety of scenes in the database.

5 Experiments

We conducted extensive experiments with the proposed method using several datasets: SUN [14] (9556 256×256 images, 522 words), LabelMe Outdoors (LMO, subset of SUN) [10] (2688 256×256 images, 33 words), ESP game image set [11] ($21,846$ images, 269 words) and IAPR benchmark [12] ($19,805$ images, 291 words). Since both LMO and SUN include dense human labeling, we use them to simulate human annotations for both training and evaluation. We use ESP and IAPR data as used by [1]. They contain user tags but no pixel labeling.

	Labeling		Tagging			
	r	$\bar{r}$	P	$\bar{P}$	R	$\bar{R}$
Makadia [1]	–	–	53.87	30.23	60.82	25.51
STF [4]	52.83	24.9	34.21	30.56	**73.83**	**58.67**
LT [5]	53.1	24.6	41.07	35.5	44.3	19.33
AP (ours)	**63.29**	**29.52**	**55.26**	**38.8**	59.09	22.62
AP-RF	56.17	26.1	48.9	36.43	60.22	24.34
AP-NN	57.62	26.45	47.5	35.34	59.83	24.01

(a) Results on LMO.

	Labeling		Tagging			
	r	$\bar{r}$	P	$\bar{P}$	R	$\bar{R}$
Makadia [1]	–	–	26.67	11.33	39.5	14.32
STF [4]	20.52	9.18	11.2	5.81	**62.04**	**16.13**
AP (ours)	**33.29**	**19.21**	**32.25**	**14.1**	47	13.74

(b) Results on SUN.

Table 1. Tagging and labeling performance on LMO and SUN datasets. Numbers are given in percentages. In (a), AP-RF replaces the GMM model in the annotation propagation algorithm with Random Forests [16]; AP-NN replaces SIFT-flow with nearest neighbor correspondences.

	ESP				IAPR			
	P	R	$\bar{P}$	$\bar{R}$	P	R	$\bar{P}$	$\bar{R}$
Makadia [1]	22	**25**	–	–	**28**	**29**	–	–
AP (Ours)	**24.17**	23.64	20.28	13.78	27.89	25.63	19.89	12.23

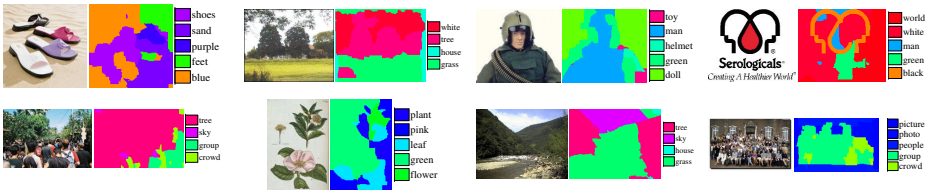
Table 2. Tagging performance on ESP and IAPR. Numbers are in percentages. The top row is quoted from [1].

We implemented the system using MATLAB and C++ and ran it on a small cluster of three machines with a total of 36 CPU cores. We tuned the algorithm's parameters on LMO dataset, and fixed the parameters for the rest of the experiments to the best performing setting: $\lambda_s = 1, \lambda_c = 2, \lambda_o = 2, \lambda_{int} = 60, \lambda_{ext} = 5$. In practice, 5 iterations are required for the algorithm to converge to a local minimum. The EM (Section 3) and inference (Section 4.2) algorithms typically converge within 15 and 50 iterations respectively. Using $K = 16$ neighbors for each image gave the best result (Fig. 6(c)), and we did not notice significant change in performance for small modifications to ϵ (Section 4.1), d (Section 3.1), nor the number of GMM components in the appearance model, M (Eq. 2). For the aforementioned settings, it takes the system 7 hours to pre-process the LMO dataset (compute descriptors and the image graph) and 12 hours to propagate annotations. The run times on SUN were 15 hours and 26 hours respectively.

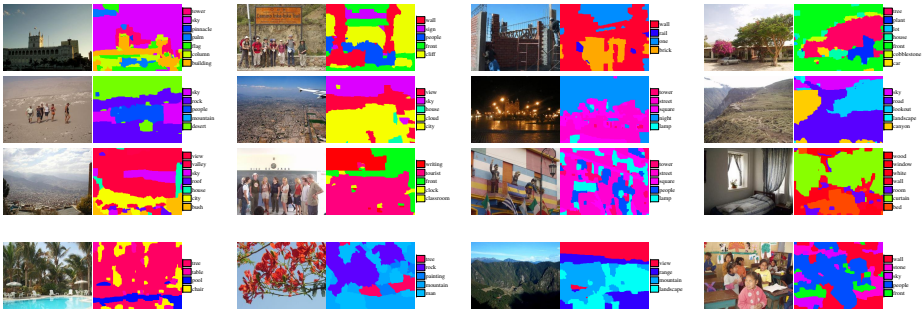
Results on SUN and LMO. Fig. 5(a) shows some annotation results on SUN using $(r_t = 0.5, r_l = 0.05)$. All images shown are initially untagged in the database. Our system successfully infers most of the tags in each image, and the labeling corresponds nicely to the image content. In fact, the tags and labels we obtain automatically are often remarkably accurate, considering that no information was initially available on those images. Some of the images contain regions with similar visual signatures, yet are still classified correctly due to good correspondences with other images. More results are available in the supplemental material.

To evaluate the results quantitatively, we compared the inferred labels and tags against human labels. We compute the global pixel recognition rate, r, and the class-average recognition rate $\bar{r}$ for images in the subset $\mathbf{I} \setminus \mathbf{I}_l$, where the latter is used to counter the bias towards more frequent words. We evaluate tagging performance on the image set $\mathbf{I} \setminus \mathbf{I}_t$ in terms of the ratio of correctly inferred tags, P (precision), and the ratio of missing tags that were inferred, R (recall). We also compute the unbiased class-average measures $\bar{P}$ and $\bar{R}$.

(a) SUN (9556 images, 522 words)



(b) ESP (21,846 images, 269 words)



(c) IAPR-TC12 (19,805 images, 291 words)

Fig. 5. Automatic annotation results (best viewed electronically). (a) SUN results were produced using ($r_t = 0.5, r_l = 0.05$) (4778 images tagged and 477 images labeled, out of 9556 images). All images shown are initially un-annotated in the database. (b-c) For ESP and IAPR, the same training set as in [1] was used, having $r_t = 0.9, r_l = 0$. For each example we show the source image on the left, and the resulting labeling and tags on the right. The word colormap for SUN is the average pixel color based on the ground truth labels, while in ESP and IAPR each word is assigned an arbitrary unique color. Example failures are shown in the last rows of (a) and (c). More results can be found in Fig. 1 and the supplemental material.

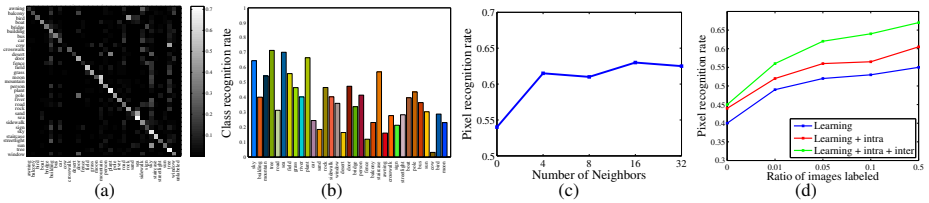


Fig. 6. Recognition rates on LMO dataset. (a) The pattern of confusion across the dataset vocabulary. (b) The per-class average recognition rate, with words ordered according to their frequency in the dataset (measured from the ground truth labels), colored as in Fig. 5. (c) Recognition rate as function of the number of nearest neighbors, K. (d) Recognition rate vs. ratio of labeled images (r_l) with different terms of the objective function enabled.

The global and class-average pixel recognition rates are 63% and 30% on LMO, and 33% and 19% on SUN, respectively. In Fig. 6 we show for LMO the confusion matrix and breakdown of the scores into the different words. The diagonal pattern in the confusion matrix indicates that the system recognizes correctly most of the pixels of each word, except for less frequent words such as *moon* and *cow*. The vertical patterns (*e.g.* in the column of *building* and *sky*) indicate a tendency to misclassify pixels into those words due to their frequent co-occurrence with other words (objects) in that dataset. From the per-class recognition plot (Fig. 6(b)) it is evident that the system generally performs better on words which are more frequent in the database.

Components of the model. Our objective function (Eq. 1) is comprised of three main components, namely the visual appearance model, and the inter-image and intra-image regularizers. To evaluate the effect of each component on the performance, we repeated the above experiment where we first enabled only the appearance model (classifying each pixel independently according to its visual signature), and gradually added the other terms. The results are shown in Fig. 6(d) for varying values of r_l. Each term clearly assists in improving the result. It is also evident that image correspondences play important role in improving automatic annotation performance. The effect of the inference module of the algorithm on the results is demonstrated visually in Fig. 3 and the supplemental material.

Comparison with state-of-the-art. We compared our tagging and labeling results with state-of-the-art in semantic labeling – Semantic Texton Forests (STF) [4] and Label Transfer [5] – and image annotation (Makadia et al. [1]). We used our own implementation of [1] and the publicly available implementations of [4] and [5]. We set the parameters of all methods according to the authors' recommendations, and used the same tagged and labeled images selected automatically by our algorithm as training set for the other methods (only tags are considered by [1]). The results of this comparison are summarized in Table 1. On both datasets our algorithm shows clear improvement in both labeling and tagging results. On LMO, $\bar{r}$ is increased by 5 percent compared to STF, and $\bar{P}$ is increased by 8 percent over Makadia et al.'s baseline method. STF recall is higher, but comes at the cost of lower precision as more tags are associated on average with each image (Fig. 7). In [5], pixel recognition rate of 74.75% is obtained on LMO at 92% training/test split with all the training images containing dense pixel labels. However, in our more challenging (and realistic) setup, their pixel recognition

rate is 53% and their tag precision and recall are also lower than ours. [5] also report the recognition rate of TextonBoost [3], 52%, on the same 92% training/test split they used in their paper, which is significantly lower than the recognition rate we achieve, 63%, while using only a fraction of the densely labeled images. The performance of all methods drops significantly on the more challenging SUN dataset. Still, our algorithm outperforms STF by 10 percent in both $\bar{r}$ and $\bar{P}$, and achieves 3% increase in precision over [1] with similar recall.

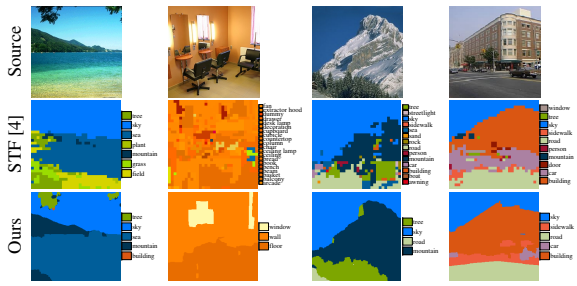


Fig. 7. Comparison with [4] on SUN dataset.

We show qualitative comparison with STF in Fig. 7 and the supplemental material. Our algorithm generally assigns less tags per image in comparison to STF, for two reasons. First, dense correspondences are used to rule out improbable labels due to ambiguities in local visual appearance. Second, we explicitly bias towards more frequent words and word co-occurrences in Eq. 4, as those were shown to be key factors for transferring annotations [1].

Results on ESP and IAPR. We also compared our tagging results with [1] on the ESP image set and IAPR benchmark using the same training set and vocabulary they used ($r_t = 0.9, r_l = 0$). Our precision (Table 2) is 2% better than [1] on ESP, and is on par with their method on IAPR. IAPR contains many words that do not relate to particular image regions (*e.g. photo*, *front*, *range*), which does not fit within our text-to-image correspondence model. Moreover, many images are tagged with colors (*e.g. white*), while the image correspondence algorithm we use emphasizes structure. [1] assigns stronger contribution to color features, which seems more appropriate for this dataset. Better handling of such abstract keywords, as well as improving the quality of the image correspondence are both interesting directions for future work.

Limitations. Some failure cases are shown in Fig. 5 (bottom rows of (a) and (c)). Occasionally the algorithm mixes words with similar visual properties (*e.g. door* and *window*, *tree* and *plant*), or semantic overlap (*e.g. building* and *balcony*). We noticed that street and indoor scenes are generally more challenging for the algorithm. Dense alignment of arbitrary images is a challenging task, and incorrect correspondences can adversely affect the solution. In Fig. 5(a) bottom row we show some examples of inaccurate annotation due to incorrect correspondences. More failure cases are available in the supplemental material.

Training set size. As we are able to efficiently propagate annotations between images in our model, our method requires significantly less labeled data than previous methods (typically $r_t = 0.9$ in [1], $r_l = 0.5$ in [4]). We further investigate the performance of the system w.r.t the training set size on SUN dataset.

We ran our system with varying values of $r_t = 0.1, 0.2, \ldots, 0.9$ and $r_l = 0.1 r_t$. To characterize the system performance, we use the F_1 measure, $F(r_t, r_l) = \frac{2PR}{P+R}$, with P and R the precision and recall as defined above. Following the user study by Vijayanarasimhan and Grauman [20], fully labeling an image takes 50 seconds on average, and we further assume that supplying tags for an image takes 20 seconds. Thus, for example, tagging 20% and labeling 2% of the images in SUN requires 13.2 human hours. The result is shown in Fig. 8. The best performance of [1], which requires roughly 42 hours of human effort, can be achieved with our system with less than 20 human hours. Notice that our performance increases much more rapidly, and that both systems converge, indicating that beyond a certain point adding more annotations does not introduce new useful data to the algorithms.

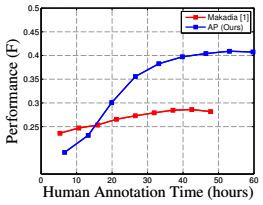


Fig. 8. Performance (F) as function of human annotation time of our method and [1] on SUN dataset.

Running time. While it was clearly shown that dense correspondences facilitate annotation propagation, they are also one of the main sources of complexity in our model. For example, for 10^5 images, each 1 mega-pixel on average, and using $K = 16$, these add $O(10^{12})$ edges to the graph. This requires significant computation that may be prohibitive for very large image databases. To address these issues, we have experimented with using sparser inter-image connections using a simple sampling scheme. We partition each image I_i into small non-overlapping patches, and for each patch and image $j \in N_i$ we keep a single edge for the pixel with best match in I_j according to the estimated correspondence $\mathbf{w}_{ij}$. Fig. 9 shows the performance and running time for varying sampling rates of the inter-image edges (*e.g.* 0.06 corresponds to using 4×4 patches for sampling, thus using $\frac{1}{16}$ of the edges). This plot clearly shows that the running time decreases *much faster* than performance. For example, we achieve more than 30% speedup while sacrificing 2% accuracy by using only $\frac{1}{4}$ of the inter-image edges. Note that intra-image message passing is still performed in full pixel resolution as before, while further speedup can be achieved by running on sparser image grids.

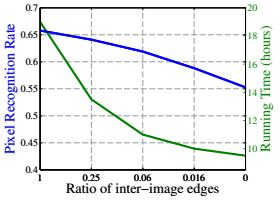


Fig. 9. Performance and run time using sparse inter-image edges.

6 Conclusion

In this paper we explored automatic annotation of large-scale image databases using dense semantic labeling and image correspondences. We treat auto-annotation as a labeling problem, where the goal is to assign every pixel in the database with an appropriate tag, using both tagged images and scarce, densely-labeled images. Our work differs from prior art in three key aspects. First, we use dense image correspondences to explicitly enforce coherent labeling (and tagging) across images, allowing to resolve visual ambiguities due to similar visual patterns. Second, we solve the problem *jointly* over the

entire image database. Third, unlike previous approaches which rely on large training sets of labeled *or* tagged images, our method makes principled use of both tag and label information while also automatically choosing good images for humans to annotate so as to optimize performance. Our experiments show that the proposed system produces reasonable automatic annotations and outperforms state-of-the-art methods on several large-scale image databases while requiring significantly less human annotations.

References

1. Makadia, A., Pavlovic, V., Kumar, S.: Baselines for image annotation. IJCV **90** (2010)
2. Wang, X., Zhang, L., Liu, M., Li, Y., Ma, W.: Arista-image search to annotation on billions of web photos. In: CVPR. (2010) 2987–2994
3. Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: ECCV. (2006) 1–15
4. Shotton, J., Johnson, M., Cipolla, R.: Semantic texton forests for image categorization and segmentation. In: CVPR. (2008)
5. Liu, C., Yuen, J., Torralba, A.: Nonparametric scene parsing via label transfer. TPAMI (2011)
6. Tighe, J., Lazebnik, S.: Superparsing: scalable nonparametric image parsing with superpixels. In: ECCV. (2010) 352–365
7. Blei, D., Ng, A., Jordan, M.: Latent dirichlet allocation. Machine Learning Research **3** (2003) 993–1022
8. Feng, S., Manmatha, R., Lavrenko, V.: Multiple bernoulli relevance models for image and video annotation. In: CVPR. (2004)
9. Liu, C., Yuen, J., Torralba, A., Sivic, J., Freeman, W.: Sift flow: dense correspondence across different scenes. In: ECCV. (2008) 28–42
10. Russell, B., Torralba, A., Murphy, K., Freeman, W.: Labelme: a database and web-based tool for image annotation. IJCV **77** (2008) 157–173
11. Von Ahn, L., Dabbish, L.: Labeling images with a computer game. In: SIGCHI. (2004)
12. Grubinger, M., Clough, P., Müller, H., Deselaers, T.: The iapr benchmark: A new evaluation resource for visual information systems. In: LREC. (2006) 13–23
13. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. (2005)
14. Xiao, J., Hays, J., Ehinger, K., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: CVPR. (2010) 3485–3492
15. Delong, A., Gorelick, L., Schmidt, F., Veksler, O., Boykov, Y.: Interactive segmentation with super-labels. In: Energy Minimazation Methods in CVPR. (2011) 147–162
16. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine Learning **63** (2006) 3–42
17. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV **42** (2001) 145–175
18. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: TOG. Volume 23. (2004) 309–314
19. Jing, Y., Baluja, S.: Visualrank: Applying pagerank to large-scale image search. TPAMI (2008)
20. Vijayanarasimhan, S., Grauman, K.: Cost-sensitive active visual category learning. IJCV (2011) 1–21