Simultaneous Multi-class Pixel Labeling over Coherent Image Sets

Paul Rivera
Research School of Computer Science
Australian National University
Canberra, ACT 0200

Stephen Gould
Research School of Computer Science
Australian National University
Canberra, ACT 0200

***Abstract*—Multi-class pixel labeling is an important problem in computer vision that has many diverse applications, including interactive image segmentation, semantic and geometric scene understanding, and stereo reconstruction. Current state-of-the-art approaches learn a model on a set of training images and then apply the learned model to each image in a test set independently. The quality of the results, therefore, depends strongly on the quality of the learned models and the information available within each training image. Importantly, this approach cannot leverage information available in other images at test time which may help to label the image at hand.**

Instead of labeling each image independently, we propose a semi-supervised approach that exploits the similarity between regions across many images in coherent image subsets. Specifically, our model finds similar regions in related images and constrains the joint labeling of the images to agree on the labels within these regions. By considering the joint labeling, our model gets to leverage contextual information that is not available when considering images in isolation.

We test our approach on the popular 21-class MSRC multi-class image segmentation dataset and show improvement in accuracy over a strong baseline model.

I. Introduction

Multi-class image labeling—the task of assigning a class label to every pixel in an image—is an important problem in computer vision [1, 2, 3, 4]. The general problem formulation can be applied to many applications, including interactive figure/ground segmentation [5], geometric and semantic scene understanding [1, 6], and stereo reconstruction [7]. For example, in the context of semantic scene understanding a common task is to annotate every pixel in the image with a label from a pre-defined set of categories, e.g., *sky*, *road*, *tree*, etc. One of the most successful approaches to these problems uses conditional Markov random fields (CRFs), which combine local information for predicting class labels (such as colour, texture and position within the image) with a prior for smoothness. The smoothness prior favours label configurations in which adjacent pixels (with similar colours) are labeled with the same category. Loosely speaking, this can be thought of as encoding the knowledge that objects have large spatial support.

Local neighbourhood priors can also encode contextual information such as co-occurrence of label pairs. In this setting confident predictions from neighbouring pixels can influence the labeling of less confident predictions, so that, *fish* adjacent to *water* is more likely than *fish* adjacent to *sky*, for example. Since the CRF is connected, this information can propagate throughout the image. More expressive forms of contextual information (such as *sky* appears above *road*) have been demonstrated by a number of researchers to improve labeling accuracy [8, 9]. The contextual information is usually derived from other regions within the same image and can therefore be limited. Moreover, since the contextual cues are derived from the single image at hand, they can sometimes reinforce incorrect interpretations of the image. Information from other images in the dataset, which could provide a rich source of context is ignored at test time in existing CRF models.

In this work, we extend the idea of a local pairwise smoothness prior between adjacent pixels within an image to that of a long-range pairwise consistency prior for propagating information between images. We then perform joint multi-class pixel labeling of all the images in the test set. Specifically, we construct a conditional Markov random field over pixels from a set of images rather than a single image. Edges between neighbouring pixels within the same image enforce the smoothness prior discussed above and that is present in many state-of-the-art approaches. Edges between pixels in different images encode our desire to label similar regions consistently across the dataset. This has the benefit of propagating contextual information from one image to another.

The following example provides some intuition into why this may be beneficial: Consider a set of images containing a variety of instances of the same object category (e.g., a car) in many of the images. In some of the images the objects may be easily recognized. However, instances from the same object category in other images may be more difficult to recognize on their own due, say, to weak local features (e.g., missing wheels) or lack of context (e.g., images of cars without visible road below the car). By finding matching regions (such as the cars' headlights) between different images we are able to exploit the more easily recognized objects to help identify the more difficult ones.

Our approach can be thought of as a semi-supervised image labeling approach. Like traditional pixel labeling approaches, we first learn a model from a set of annotated training images. However, instead of using that model to label each image in the dataset in isolation, we enforce soft labeling constraints between regions with similar appearance in different images. Importantly, the soft constraints are found in an unsupervised

manner (i.e., without knowing the class labels). Experiments on the 21-class MSRC dataset [10] demonstrate that this leads to an improvement over a strong baseline CRF model.

II. Background and Related Work

Our work builds on the work from a number of researchers who have investigated the problem of multi-class pixel labeling. Perhaps, the most influential works are those of He et al. [1] and Shotton et al. [2], which are early examples of the use of conditional Markov random fields (CRFs) for the multi-class pixel labeling task. In particular, these models define a grid-structured pairwise CRF over pixels with smoothness prior. Our work extends this approach from inference on single images to concurrent inference on multiple images in a set.

A line of research known as "co-segmentation" has also studied the problem of labeling pixels in multiple images simultaneously in recent years, with great success [11, 12]. Here the task is to perform joint segmentation of the same, or a similar looking object, from two or more images. The assumption is that using more images provides additional information that can help improve the segmentation quality—an assumption that is supported by impressive experimental results (see, for example, Rother et al. [11]).

Similar to co-segmentation, our work does joint image segmentation over sets of images. However, our work has a number of key differences. First, we do not impose the constraint of having a common object in the sets of images. Instead, we focus on the semantic classes of the objects appearing in the images. For instance, in a collection of images that all have a car in them, we do not require the cars to be exactly the same. Second, we do not assume *a priori* that the images will contain the same object. Instead, we employ a matching stage to correspond similar regions and reason that these imply similar semantics. Last, we are interested in the case of multi-class labeling rather than figure-ground segmentation. That is, we may have multiple different objects and background regions in the image set, and wish to label each of these. To achieve this, we use a local region matching algorithm [13] to form correspondences between the similar regions of these images that are encoded in a CRF defined over the entire image set.

Our work makes use of the observation that scenes can be clustered into similar types, and that by finding regions with similar appearance within different images we can constrain labelings between images. To cluster the images we compute a gist descriptor [14] and perform hierarchical agglomerative clustering (see Section III-C below). Other techniques for building similarity graphs over image collections have been explored in the literature (e.g., [15]), however, the aim in these works is often for image categorization or navigation, not to provide context for pixel labeling.

Other recent works have exploited this observation for scene labeling, but for the purpose of labeling a single image. For example, Liu et al. [16] aligns a novel scene with similar scenes from a large corpus of labeled images and transfer the labels from the corpus to the novel scene. The algorithm performs well on background classes and can be easily expanded to incorporate new objects. However, since the alignment is done at a scene level small objects are often missed. Moreover, unlike our approach, it does not make use of the similarity between images in the set of images to be labeled.

III. Pixel Labeling for Image Sets

In this section we describe our approach to dataset labeling. Unlike traditional approaches to multi-class pixel labeling, which learn a model and then apply the learned model to each test image in isolation, our method simultaneously labels collections of images thereby leveraging contextual information available from different images at test time. We begin by describing a typical conditional Markov random field formulation of the single-image pixel labeling problem. We then extend this formulation to the case of multiple images.

A. Conditional Markov Random Fields for Pixel Labeling

Conditional Markov random fields (CRFs) are a class of probabilistic models for encoding conditional probability distributions over correlated random variables. They were first introduced as a generalization to Markov random fields (MRFs) by Lafferty et al. [17] for language modeling, but have subsequently proven to be a powerful framework for many problems in computer vision such as multi-class pixel labeling [1, 2].

Given an image $\mathcal{I}$, a CRF for pixel labeling defines an *energy function* over different label configurations where lower energy labelings are preferred by the model.[1] Concretely, let $\mathcal{L}$ be a set of discrete labels, e.g., $\{\text{sky}, \text{road}, \ldots\}$, and let $\boldsymbol{y} = (y_1, \ldots, y_n)$ be a vector of labels for the image where $y_p \in \mathcal{L}$ is the label assigned to pixel p. A pairwise CRF defines an energy function as the combination of *unary* and *pairwise potentials* as

$$E(\boldsymbol{y}; \mathcal{I}) = \sum_{i=1}^{n} \psi_i(y_i; \mathcal{I}) + \sum_{ij \in \mathcal{N}_8} \psi_{ij}(y_i, y_j; \mathcal{I}) \qquad (1)$$

where $\psi_i(y_i; \mathcal{I})$ are the unary potentials defined for each variable and $\psi_{ij}(y_i, y_j; \mathcal{I})$ are the pairwise potentials defined over adjacent variables in the image. Here $\mathcal{N}_8$ represents the 8-connected neighbourhood of pixels, i.e., the subset of pairs (i, j) such that pixels i and j and adjacent to each other in the image.

The unary potentials capture an individual pixel's preference for each label given some local features. In our work, we construct the local features as follows: First, we convolve the image with a 17-dimensional filter bank to produce raw image features. The specification for the filter bank is described in Shotton et al. [2]. Next, we define a 3×3 grid of cells centered around each pixel p, where each grid cell covers a 5×5 patch of pixels, and compute the mean and standard deviation of raw (17-dimensional) features in each cell. Finally, we append the

[1] Formally, the energy function is defined as the negative log of the unnormalized conditional probability, i.e., if $E(\boldsymbol{y}; \boldsymbol{x})$ is the energy function, then $P(\boldsymbol{y} \mid \boldsymbol{x}) \propto \exp\{-E(\boldsymbol{y}; \boldsymbol{x})\}$.

raw image features, the mean and standard deviation features and the normalized x and y location of the pixel together into a 325-dimensional local feature vector.

The local features are used to build a classifier that estimates the probability of each label given the features. These will ultimately be used for specifying the unary potentials in our model. To learn this multi-class classifier, we first train a one-versus-all boosted decision tree classifier [18] for each label $\ell \in \mathcal{L}$. We then combine the output of these one-versus-all classifiers through multi-class logistic regression trained via maximum-likelihood to calibrate the scores [19]. Concretely, let $f_p \in \mathbb{R}^{325}$ be the local feature vector for pixel p. We learn a boosted classifier $\phi_\ell : \mathbb{R}^{325} \to \mathbb{R}$ for each class $\ell \in \mathcal{L}$. Let $\boldsymbol{\phi}(f_p) = (\phi_1(f_p), \ldots, \phi_L(f_p))$ be the vector of scores from the boosted classifiers. Our multi-class classifier is then

$$P(y_p = \ell \mid f_p) = \frac{\exp\left\{\boldsymbol{\theta}_\ell^T \boldsymbol{\phi}(f_p)\right\}}{\sum_{k\in\mathcal{L}} \exp\left\{\boldsymbol{\theta}_k^T \boldsymbol{\phi}(f_p)\right\}} \tag{2}$$

where $\boldsymbol{\theta}$ are the learned parameters. The unary potential for each pixel is formed by taking the negative log-probability for each class from this logistic classifier,

$$\psi_i(y_i; \mathcal{I}) = -\log P(y_i \mid f_i). \tag{3}$$

While our model makes use of boosted decision tree classifiers and multi-class logistic regression, the quality of the final model appears quite robust to this choice and other researchers have reported similar baseline results using other classifier architectures, such as random forests and support vector machines.

The pairwise potential imposes a contrast-sensitive smoothness prior [5]. In other words, the model prefers configurations where adjacent pixels take the same label. More formally, we define the contrast-sensitive smoothness prior for two adjacent pixels i and j as

$$\psi_{ij}(y_i, y_j; \mathcal{I}) = \begin{cases} \frac{\lambda_1}{d_{ij}} + \frac{\lambda_2}{d_{ij}} \exp\left\{-\frac{\|x_i - x_j\|^2}{2\beta}\right\} & y_i \neq y_j \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where x_i and x_j are the RGB colour vectors for pixels i and j, respectively, and $\beta = \left\langle \|x_i - x_j\|^2 \right\rangle_{ij\in\mathcal{N}_8}$ is the mean-square-difference in colour over all adjacent pixels in the image. The non-negative parameters λ_1 and λ_2 weight the prior relative to the unary terms and are learned by cross-validation on the training set of images to maximize overall pixel accuracy. Here d_{ij} scales the contribution of the prior by the distance between the pixels, i.e., $d_{ij} = 1$ for 4-connected pixels, and $d_{ij} = \sqrt{2}$ for diagonally-connected pixels.

The contrast-sensitive pairwise potentials capture our belief that images are generally smooth with label changes only occurring at the boundary between regions of different appearance. While this assumption is generally a good one, its implementation in CRFs for pixel labeling suffers from a number of drawbacks. First, correct labeling relies on good local evidence, i.e., local features that can predict the correct category label. Second, determining where the region boundaries are can be difficult when only considering differences in colour between neighbouring pixels. Last, contextual information that may assist the correct labeling of a region may not be present in all images. Extending our model to image sets rather than individual images partially addresses these drawbacks.

B. Extending Conditional Random Fields to Image Sets

Consider two images with similar objects appearing in each of the images. Now assume that we are given correspondence information between the images, i.e., we are told that a subset of pixels in one of the images corresponds (semantically) to a subset of pixels in the other image. Then, just like the pairwise smoothness prior, we could encode a soft constraint that these pixels be labeled as belonging to the same semantic class. Note, that we do not need to be told what the semantic class is for this information to be useful. Furthermore, the correspondence information that we receive may contain errors so we will want our constraint to be weighted by our confidence in the correspondence.

Formally, let $\mathcal{I}_1$ and $\mathcal{I}_2$ be two images and let $\mathcal{P} = \{(p_1, p_2) : p_1 \in \mathcal{I}_1,\, p_2 \in \mathcal{I}_2\}$ be a set of correspondences between pairs of pixels (p_1, p_2) from the first and second image, respectively. We can now define a joint image labeling energy function as

$$E(\boldsymbol{y}_1, \boldsymbol{y}_2; \mathcal{I}_1, \mathcal{I}_2) = E(\boldsymbol{y}_1; \mathcal{I}_1) + E(\boldsymbol{y}_2; \mathcal{I}_2) + \sum_{(p,q)\in\mathcal{P}} \psi_{pq}(y_{1,p}, y_{2,q}; \mathcal{I}_1, \mathcal{I}_2) \tag{5}$$

where the last term encodes our soft constraint that the labels for corresponding pixels, p in the first image and pixel q in the second image, should match. Specifically we have,

$$\psi_{pq}(y_{1,p}, y_{2,q}; \mathcal{I}_1, \mathcal{I}_2) = \begin{cases} \lambda_3 c_{pq} & y_{1,p} \neq y_{2,q} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $y_{1,p}$ and $y_{2,q}$ are the labels for pixel p in the first image and pixel q in the second image, and c_{pq} is a (non-negative) score that represents our confidence in the match. In the following section we will describe how the matches and confidence scores are obtained. The constant λ_3 weights this between-image constraint against the unary and pairwise terms from the within-image components of the model, i.e., $E(\boldsymbol{y}_1; \mathcal{I}_1)$ and $E(\boldsymbol{y}_2; \mathcal{I}_2)$.

Clearly this idea can be extended to multiple images where we define an energy function $E(\{\boldsymbol{y}_i\}_{i=1}^n; \{\mathcal{I}_i\}_{i=1}^n)$ over the joint labeling of multiple images $\mathcal{I}_1, \ldots, \mathcal{I}_n$, and let the set $\mathcal{P}$ contain pairs of corresponding pixels between any two of images in the set.

C. Matching Regions Between Images

Our model requires that we find good matches between regions in different images. To reduce computational requirements while still finding matches that are likely to help improve labeling accuracy we adopt a two stage approach. In the first stage we compute a global gist descriptor [14] for each image. We then cluster the images (separately for the training

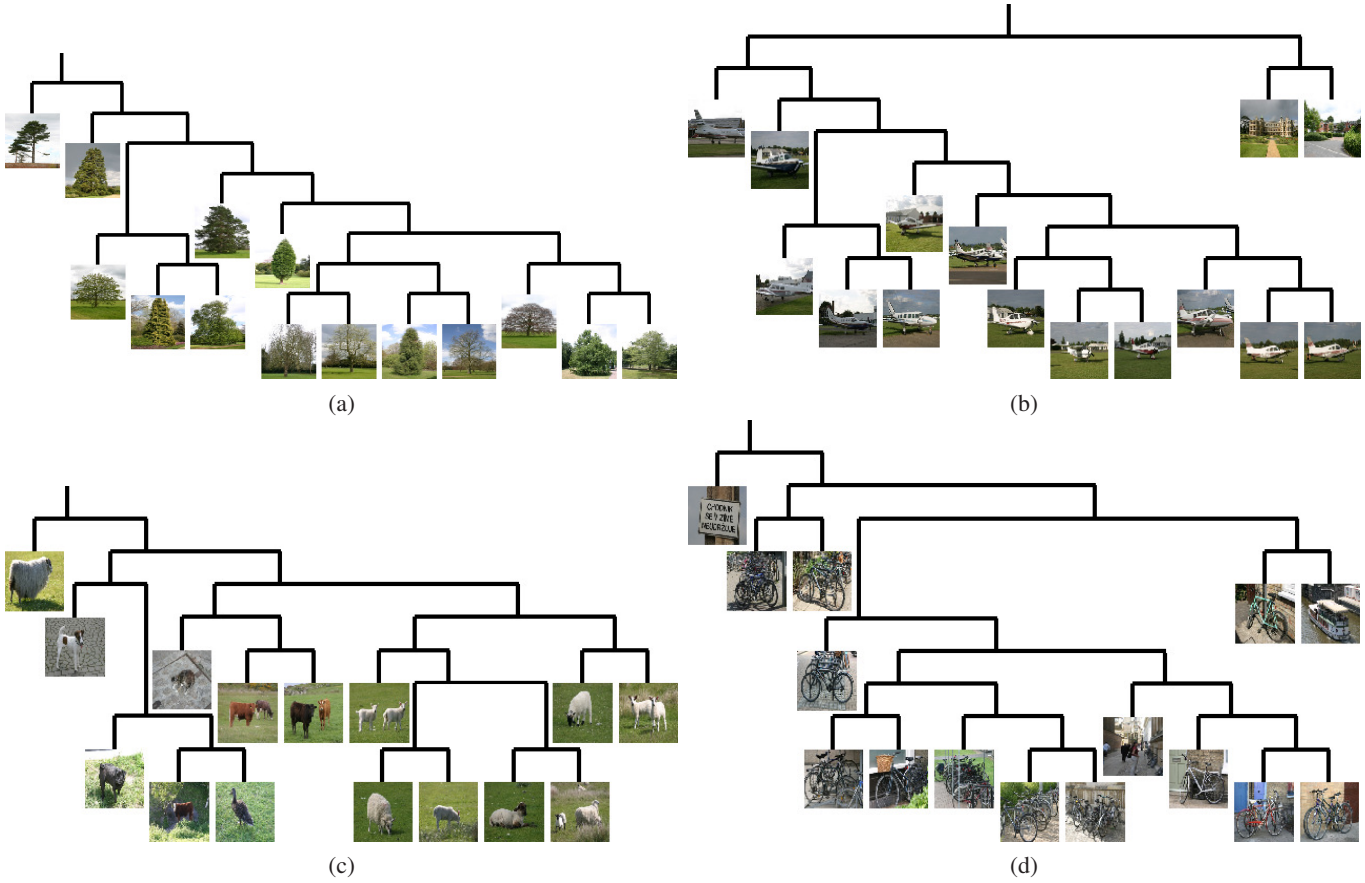


Fig. 1. Some example clusters obtained by running hierarchical agglomerative clustering on gist features. Scenes with similar objects tend to be grouped together. For example, (a) contains exclusively trees, while (b) is mostly aeroplanes. The tree structure represents merge order during clustering. See text for details.

and testing sets) by performing hierarchical agglomerative clustering as follows: Let $g_i \in \mathbb{R}^n$ be the gist descriptor for the i-th image in image set $\mathcal{S}$ (either training or testing), let $D_{ij} = \|g_i - g_j\|_2$ be the distance between two images i and j in gist-space, and let $N^{\max}$ be the maximum number of images that we can tolerate per cluster.[2] Initially we start with each image in its own cluster. We then repeatedly merge two clusters at a time until any further merge results in a cluster of size greater than $N^{\max}$. At each iteration we find the two clusters such that their combined size is less than $N^{\max}$ and with minimum distance between their elements. Formally, we find clusters $\mathcal{S}_a$ and $\mathcal{S}_b$ satisfying

$$(a, b) = \underset{a \neq b : |\mathcal{S}_a \cup \mathcal{S}_b| \leq N^{\max}}{\operatorname{argmin}} \left\{ \min_{i \in \mathcal{S}_a, j \in \mathcal{S}_b} D_{ij} \right\} \tag{7}$$

We then merge the clusters to create a new cluster $\mathcal{S}_a \cup \mathcal{S}_b$. Some example clusters on the MSRC [10] dataset for $N^{\max} = 15$ are shown in Figure 1.

In the second stage, we look for similar regions between images within the same cluster. We use the PATCHMATCH algorithm introduced by Barnes et al. [20]. Briefly, PATCHMATCH is an approximation algorithm that performs an incremental search over all patches in one image to find the most similar patch in another image with respect to some distance metric. The algorithm takes two images $\mathcal{I}_1$ and $\mathcal{I}_2$, and the patch size as input. Since we are interested in matching every patch in the image, the output of the algorithm can be thought of as an approximate *nearest-neighbour field* (NNF) which is defined as a function $f_{1\to2} : \mathcal{I}_1 \to \mathcal{R}_2$ where $\mathcal{R}_2$ gives the offset of the approximated most similar patch in image $\mathcal{I}_2$. Similarly, the algorithm will produce $f_{2\to1} : \mathcal{I}_2 \to \mathcal{R}_1$.

The PATCHMATCH algorithm has three phases. The first phase initializes the target NNF with random offsets. Most of these initial assignments are likely to be bad matches, however, some of the matches will be good. The second phase exploits these good matches by propagating them to neighbouring patches under the assumption that images have a naturally smooth structure—i.e., a neighbouring patch of p will probably be a good match for a neighbouring patch of $f(p)$. The third phase performs a random search over patches within a radius of the best offsets found so far for potentially better matches. PATCHMATCH is essentially a local search and the third phase allows it to escape from local maxima. The algorithm iterates over the second and third phases and terminates after a fixed number of iterations or after convergence (no improved matches can be found).

[2]See the experimental section for metrics on running time and memory usage as a function of the number of images per cluster.

The randomized search strategy for finding an approximate NNF instead of an exact NNF also allows PATCHMATCH to run very efficiently. Furthermore, the quality of the matches produced by PATCHMATCH have been shown to be good and the algorithm has been successfully used in image editing tasks such as re-targeting, completion, and reshuffling [13, 20, 21].

We run the PATCHMATCH algorithm on all pairs of images in each cluster and retain the top three nearest-neighbours per image patch. These are then used to add edges between images when constructing our image-set CRF (as described in Section III-B). This achieves our goal of encoding the soft constraint that regions with similar appearance in different images should be labeled the same. Figure 2 shows examples of good and bad matches found by the PATCHMATCH algorithm. Here we define "bad" to mean matches for which the corresponding regions have different semantics (even though they may have similar appearance). We note that our algorithm currently searches over patches of the same size and orientation. Extending the search to be scale and rotation invariant is an interesting topic for future work.

Since matches for some patches may not be found and to make our model more robust to poor matches we adjust the strength of the between-image label constraint as a function of the quality of the match as indicated by c_{pq} in Equation 6. Specifically, let s_{pq} be the score returned by the PATCHMATCH algorithm for matching pixel p in image $\mathcal{I}_1$ to pixel $q = f(p)$ in image $\mathcal{I}_2$, where a lower score indicates a better match. Then we set c_{pq} to

$$c_{pq} = \exp\left\{-\frac{s_{pq}}{2\beta}\right\} \tag{8}$$

where β is the mean match score returned by PATCHMATCH for all pairs of matches in the image set.

D. Inference

After constructing the CRF to add soft constraints between images, we run inference to find the most likely joint labeling of all images in the image set. However, exact inference in our model—as well as the baseline CRF model—is intractable and we have to resort to an approximate inference scheme.

By design, our energy function belongs to the class of so-called *regular* (or submodular) energies [22] and can therefore be minimized using the α-expansion variant of graph-cuts [23, 24]. The α-expansion algorithm is a move-making algorithm that solves a series of binary problems in an iterative manner. The variables in the model are initialized to some valid assignment—in our case we take the minimizing assignment from each unary term. Then at each iteration, the optimal assignment is found in the sub-space of labels where each variable can either keep its current assignment or switch to the label $\alpha \in \mathcal{L}$. Since our energy function is submodular this can be done exactly. A new label $\alpha \in \mathcal{L}$ is then chosen for the next iteration and the procedure repeated until no move results in a lower energy assignment.

For computer vision applications where the number of pairwise terms in the energy function is sparse, very efficient algorithms exist for solving the resulting optimization problem at each α-expansion iteration (see [24]). While our energy function has more pairwise terms than the standard 8-connected CRF they are still relatively few compared to the number of variables in the model. As discussed above, we limit the number of matches from each pixel to three (although we place no limit on the number of matches that can be made *to* a pixel). As such, we found inference to be quite fast and in practice followed a linear increase in running-time as the number of images increased (see Figure 6).

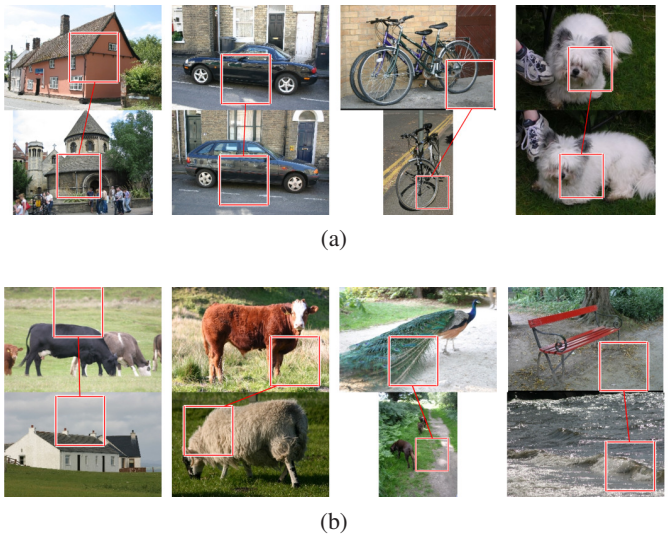


Fig. 2. Some examples of matches produced by PATCHMATCH between 100×100 patches. Images in (a) show some of the good matches. Images in (b) show some of the bad matches. We mitigate the effect of these bad matches using the match confidence score, which is determined by the RGB distance between the patches. Although, there are some cases where a match between two different objects have a very low RGB distance and, consequently, a very high confidence score.

IV. EXPERIMENTAL RESULTS

We conducted experiments on the multi-class image labeling task and compared the results of using CRFs on image sets, with correspondences over similar regions, and CRFs without these correspondences, i.e., the baseline model on individual images. The dataset used for these experiments was the 21-class MSRC dataset [10] consisting of 591 annotated images. Each image is approximately 320×240 and the dataset contains a *void* label for unknown pixels, which are ignored during both training and evaluation.[3] As is standard on this dataset, we split the images into a training set consisting of 315 images and an evaluation set consisting the remaining 276 images. We have established five different folds of the dataset where each fold has a random partition of the images into training and test sets. These sets were then divided into clusters of up to $N^{\max} = 15$ images as described in Section III-C above.

[3]The dataset also contains labels for *mountain* and *horse*. However, there are very few instances of these and we follow the literature in treating these categories as *void*.

TABLE I
RESULTS FOR 100×100 PATCHES

	Overall class accuracy		Average class accuracy	
Fold	Baseline	Our model	Baseline	Our model
1	73.2	**74.1**	58.0	**58.7**
2	76.1	**77.4**	64.1	**65.0**
3	71.6	**73.4**	59.3	**60.6**
4	72.4	**73.3**	**58.0**	57.8
5	75.9	**76.0**	**63.2**	63.0

A comparison of the overall class accuracy and average class accuracy of the baseline and our model using patch matches of size 100×100 pixels.

TABLE II
RESULTS FOR 25×25 PATCHES

	Overall class accuracy		Average class accuracy	
Fold	Baseline	Our model	Baseline	Our model
1	73.2	**73.8**	**58.0**	57.8
2	76.1	**77.1**	64.1	**65.0**
3	71.6	**72.8**	59.3	**59.9**
4	72.4	**72.9**	**58.0**	57.7
5	75.9	**76.2**	63.2	**63.5**

A comparison of the overall class accuracy and average class accuracy of the baseline and our model using patch matches of size 25×25 pixels.

All parameters in the model were learned on the set of training images. Specifically, we used a random sample of pixels from the training images to learn the unary terms and then performed cross-validation to find the best weights λ_2 and λ_3, for the within-image pairwise smoothness term and between-image label matching constraint, respectively (we found that λ_1 had little effect on performance and simply set it to zero for all experiments). Note that the learned parameter for λ_2 was different for the baseline model and our model with between-image terms.

In the region matching procedure discussed in Section III-C, we have chosen RGB as our colour-space and used the sum-of-squared difference over the RGB channels as our distance metric in the PATCHMATCH algorithm. Furthermore, we partitioned each image into non-overlapping patches on a regular grid. For each grid location in a particular image, we find the best matches over all possible patches of other images in the cluster using the PATCHMATCH algorithm. Thus, in a cluster with N images, this process produces $N-1$ matches for each grid location. To reduce computational complexity and prune poor matches, we limit the between-image edges to the top three matches (based on the PATCHMATCH score) per patch.

We measure performance by two different metrics and report average results on the evaluation set for each fold. The first performance metric measures overall accuracy and is simply the proportion of correctly labeled pixels. The second performance metric is the class-averaged performance, which is normalized for the different abundance of classes in the dataset. Here we separately compute the proportion of each class labeled correctly and average the result.

Table I and Table II show the quantitative results from our experiments using patches of dimensions 100×100 and 25×25 pixels, respectively. Inclusion of the soft between-image constraint improves the overall class accuracy by an average of 1.0% and the average class accuracy by 0.5% for the 100×100. While using 25×25 gives a slightly lower average improvement of 0.7% for the overall class accuracy and 0.3% for the average class accuracy. Our results show a consistent improvement in overall accuracy in all folds and improvement in class-averaged accuracy in most of the folds.

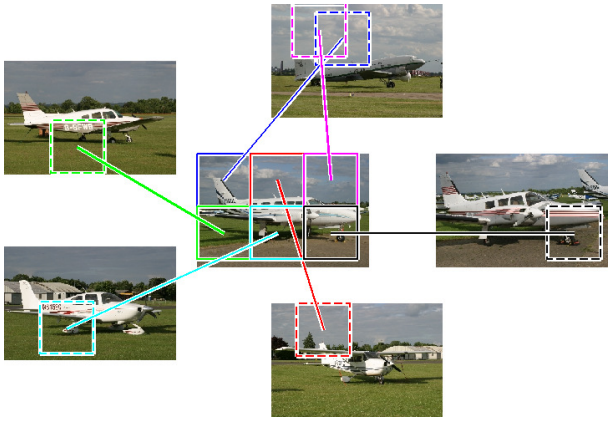


Fig. 3. **Best viewed in colour.** Example matches for patches sizes of 100×100 at fixed grid locations. Shown are the top matches for each grid location in the center image to other images in the same cluster. Matches from other images in the cluster to the center image are not shown.

We also show the qualitative results to see how the soft constraint over similar regions affect the labeling of the images. In Figure 4, we show some of the images with improved labelings using our model. On the other hand, our model has also introduced some degradation to the labeling accuracy. We show some of the degraded labelings in Figure 5.

Figure 3 shows the top matching regions for one of the images in our dataset. Clearly all of the matches are in the correct context (i.e., aeroplanes on a field). This is helped by our clustering algorithm, which groups similar images together based on global structure. Moreover, a number of the matches are consistent with respect to the semantics of the corresponding pixels, for example, the front of the aeroplane (bottom-right). However, there are also matches which are partially inconsistent, e.g., the aeroplane's tail (top-left). This explains why performance sometimes degrades and suggests that a more robust pairwise potential, which only requires a subset of the pixels within each pair of matched regions to agree, may improve performance.

Finally, Figure 6 shows that the running time and memory usage of our algorithm grow linear to the number of images in the cluster. We have chosen a maximum of 15 images per cluster in our experiments. At this size, it seems to already have a reasonable number of matching images within the cluster while keeping the running time below 7 minutes and the memory usage below 1GB. For larger sets inference can become intractable and we are currently investigating ways to reduce some of the computational overhead, e.g., by limiting the total number of edges per variable or using other inference algorithms.

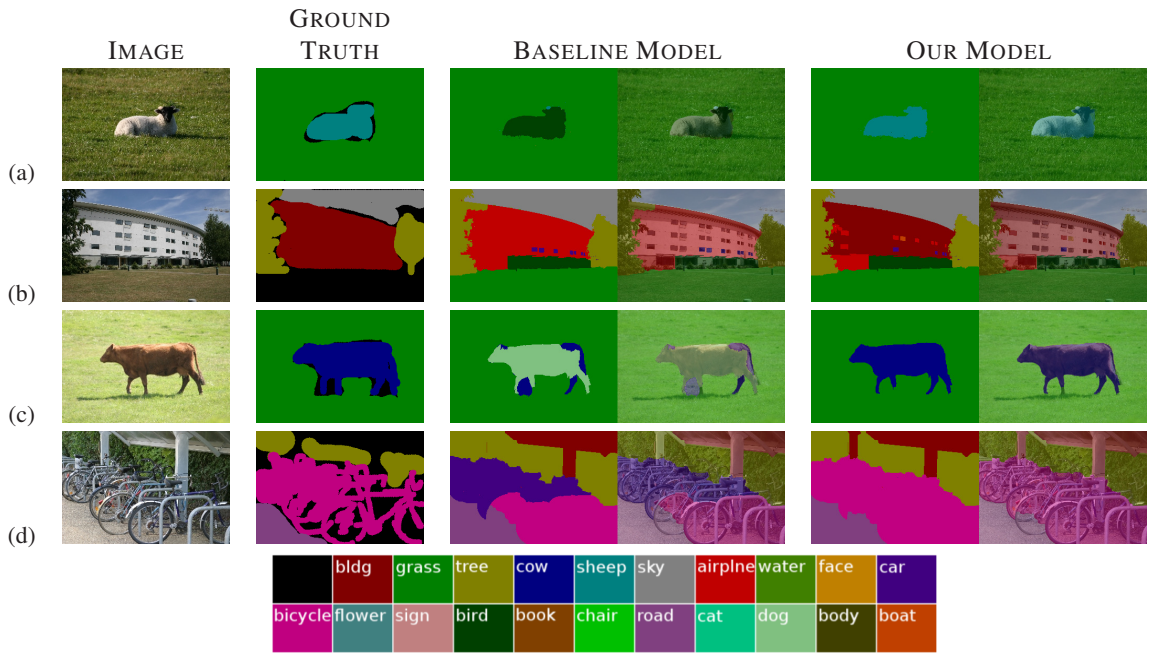
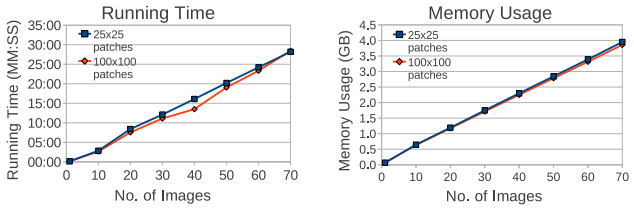


Fig. 4. **Best viewed in colour.** These images show improved pixel labeling accuracy when using our model. The images in row (a) show the labeling of the sheep gets corrected from a baseline model labeling of bird using our model. Row (b) shows that the building was labeled as an airplane using the baseline model and that its labeling gets corrected using our model. Rows (c) and (d) show significant improvements in the labeling of the cow and bicycles, respectively.



(a) Running Time (b) Memory Usage

Fig. 6. Running time and memory consumption as a function of the size of the image set for joint segmentation using patch sizes of 25×25 and 100×100. Both show a linear increase with increase in number of images.

V. Discussion

This work introduces a novel approach to the problem of multi-class pixel labeling. Instead of treating each image in the test set as an isolated test case, we find regions with similar appearance between images and prefer solutions where these regions are labeled consistently across the dataset. The advantage of this approach is that contextual information can propagate through all images in a collection thereby improving overall accuracy across the images.

Our research suggests a number of directions for future work. First, there are a number of meta-parameters—patch size, image and patch similarity metric, etc.—which effect the performance of our method and a detailed exploration of these parameters may lead to greater gains from our approach. One important meta-parameter, governed by computational and memory constrains, is the size of the sets over which we can perform tractable inference. Expanding to larger image sets necessitates distributed algorithms for multi-label energy minimization. Such approaches have been explored in the case of graphs with regular structure [25], but it is unclear whether these are appropriate to the less regular structures admitted by our between-image constraints.

Second, our matches are currently constrained to be of the same size and orientation. Generalizing the matches to be scale and rotation invariant would almost certainly result in some better matches. Furthermore, analysis of the matches (e.g., see Figure 3) suggests that enforcing that all corresponding pixels within the matched regions agree may contribute to degradation in performance in some cases. Constructing a more robust constraint, such as only requiring a subset of the pixels to agree, would avoid this issue and is related to current active research in higher-order potentials for Markov random fields.

Third, we would like to explore model-free approaches where instead of using the training data to learn appearance models for each of the classes of interest, we could simply add soft constraints between images in the training set and images in the test set. Unlike the constraints we have now between two images in the test set, these additional constraints would allow for labels to be transferred from the training set. This could have a number of advantages for large-scale systems, such as the ability to incrementally grow the training set without having to re-learn the model parameters.

Last, there will always be vastly more unlabeled images than images with available annotations. Our current work suggests that there exist opportunities to exploit these images for constraining label configurations and improving scene understanding. We are excited about exploring other ways in

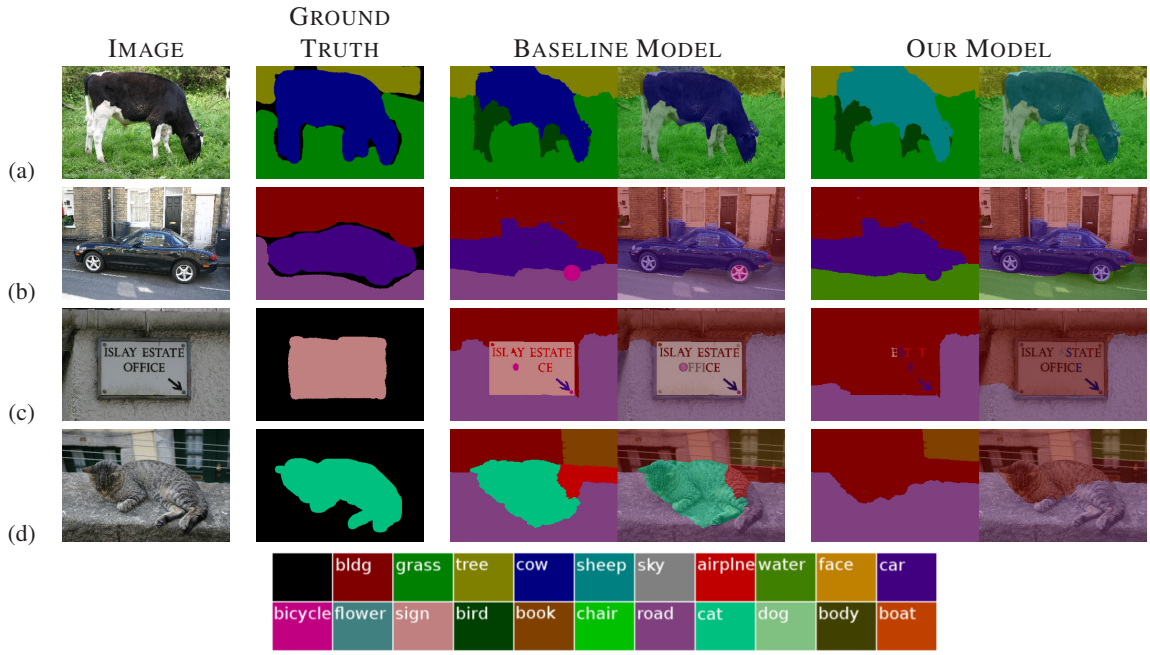


Fig. 5. **Best viewed in colour.** These images show some degradation in the pixel labeling accuracy when using our model. Row (a) shows how the labeling of a cow switches to sheep with our model. Row (b) shows how the road gets incorrectly labeled as sea with our model. Rows (c) and (d) show how a sign and a cat both get incorrectly labeled as building with our model.

which considering image collections jointly can help to produce better interpretations for all the images in the collection.

ACKNOWLEDGMENT

This work was supported by the NCI National Facility at the Australian National University.

REFERENCES

[1] X. He, R. Zemel, and M. Carreira-Perpinan. Multiscale conditional random fields for image labeling. In *CVPR*, 2004.

[2] J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

[3] L. Ladicky, C. Russell, P. Kohli, and P.H.S. Torr. Associative hierarchical CRFs for object class image segmentation. In *ICCV*, 2009.

[4] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

[5] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*, 2001.

[6] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 75(1):151–172, 2007.

[7] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 2002.

[8] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, 2007.

[9] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller. Multi-class segmentation with relative location prior. *IJCV*, 80(3): 300–316.

[10] A. Criminisi. Microsoft research cambridge (MSRC) object recognition pixel-wise labeled image database (version 2), 2004.

[11] C. Rother, V. Kolmogorov, T. Minka, and A. Blake. Cosegmentation of image pairs by histogram matching—incorporating a global constraint into MRFs. In *CVPR*, 2006.

[12] S. Vicente, C. Rother, and V. Kolmogorov. Object cosegmentation. In *CVPR*, 2011.

[13] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein. The generalized PatchMatch correspondence algorithm. In *ECCV*, 2010.

[14] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42(3): 145–175, 2001.

[15] K. Heath, N. Gelfand, M. Ovsjanikov, M. Aanjaneya, and L. J. Guibas. Image webs: Computing and exploiting connectivity in image collections. In *CVPR*, 2010.

[16] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing: Label transfer via dense scene alignment. In *CVPR*, 2009.

[17] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[18] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.

[19] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, pages 61–74, 1999.

[20] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. In *SIGGRAPH*, 2009.

[21] H. Zhang and L. Quan. Partial similarity based nonparametric scene parsing in certain environment. In *CVPR*, 2011.

[22] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26:65–81, 2004.

[23] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *ICCV*, 1999.

[24] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26:1124–1137, 2004.

[25] A. Delong and Y. Boykov. A scalable graph-cut algorithm for N-D grids. In *CVPR*, 2008.