Hybrid learning of large jigsaws

Julia Lasserre
Department of Engineering
University of Cambridge, UK
jal62@cam.ac.uk

Anitha Kannan, John Winn
Microsoft Research
Cambridge, UK
{ankannan,jwinn}@microsoft.com

Abstract

A jigsaw is a recently proposed generative model that describes an image as a composition of non-overlapping patches of varying shape, extracted from a latent image. By learning the latent jigsaw image which best explains a set of images, it is possible to discover the shape, size and appearance of repeated structures in the images. A challenge when learning this model is the very large space of possible jigsaw pixels which can potentially be used to explain each image pixel. The previous method of inference for this model scales linearly with the number of jigsaw pixels, making it unusable for learning the large jigsaws needed for many practical applications. In this paper, we make three contributions that enable the learning of large jigsaws - a novel sparse belief propagation algorithm, a hybrid method which significantly improves the sparseness of this algorithm, and a method that uses these techniques to make learning of large jigsaws feasible. We provide detailed analysis of how our hybrid inference method leads to significant savings in memory and computation time. To demonstrate the success of our method, we present experimental results applying large jigsaws to an object recognition task.

1. Introduction

Many computer vision applications such as object recognition, stereo matching and image segmentation use patch-based representation for modelling the appearance of objects or object parts (cf. [1, 8, 5]). These patch-based models typically use fixed shaped patches and so they suffer from mismatch between the object shape and the patch shape. For example, they must model the variability of any background regions included in the patch. When the shape of the patch is *learned*, it not only overcomes this mismatch problem, but also improves recognition since the shape of the patch is also informative as to the identity of the object. In [10], a generative model was presented that describes an image as non-overlapping composition of patches extracted from a latent image known as a jigsaw. This jigsaw model is capable of learning the shape of the patches so as to match the shape of repeated structures in the image. As shown in [10], such repeated structures are highly correlated with object parts.

A limitation of the inference method proposed in [10] is that the time and memory requirements scale linearly with the number of pixels in the latent jigsaw image. This prohibits the use of this model for learning the large jigsaws needed for many practical applications, such as object recognition and image segmentation. In this paper, we propose a hybrid learning method which is able to learn jigsaws of much greater size within the same memory/time constraints. In particular, we make three main contributions that allow the learning of large jigsaws from sizeable collections of images:

- a sparse belief propagation algorithm for inferring the mapping from the image to the jigsaw.
- a hybrid generative/discriminative method, Hybrid BP, that significantly increases the sparseness of messages propagated during belief propagation, especially when the jigsaw size is large.
- an effective method for learning large jigsaws which exploits the memory and time savings given by Hybrid BP.

The paper is organized as follows: in section 2, we describe previous work on sparse and hybrid methods. Section 3 describes the jigsaw generative model. Section 4.1 describes our sparse belief propagation method, whilst, section 4.2 explains our hybrid belief propagation method and details the resulting savings in memory and computation time. In Section 5, we present a jigsaw learning method, based on hybrid belief propagation, which is capable of learning large jigsaws. We also present experimental results applying large jigsaws to an object recognition task that demonstrates the suitability of the jigsaw model and the success of our hybrid learning method.

1-4244-1180-7/07/$25.00 ©2007 IEEE

2. Related Work

The closest work to jigsaw model is the epitome model for image patches, or an image [9]. Unlike jigsaws, epitomes use a set of fixed shaped and sized patches and provide only a model of image patches rather than entire images, unless patch averaging is used. For a more detailed qualitative and quantitative comparison of the two models, see [10].

Markov random fields (MRFs) are widely used as a probabilistic model for capturing local interactions. For MRFs with variables that have small state space (*i.e*. each variable can have one of a small set of values), inference can be performed efficiently using methods such as graph cuts and belief propagation. However, when the state space of the variables is large, these methods run into memory or speed limitations. Previous approaches for overcoming these limitations have involved either pruning the state space of a variable based on local evidence [4, 11] or pruning the messages sent in belief propagation using a sparse representation [14]. In section 4.1 we will describe these previous methods in more detail and introduce a variant of the latter approach for performing inference in the jigsaw model.

The term *hybrid* is used in a variety of contexts. In probabilistic model learning, hybrid refers to learning by exploiting a combination of generative and discriminative approaches. One typical method is to use a convex combination of a discriminative and generative log likelihood function, where the regime for best predictive performance lie between the two extremes [2]. In this context, [12] derives a rigorous method for discriminatively training a generative model. Alternatively, in [6] a combination of a carefully constructed generative model and a general purpose feedforward classifier is used to rapidly infer motor programs for digit images. In [7], a generative model is proposed that generates both the data and their labels. In this paper, we use a discriminative model for the sole purpose of finding a good sparse approximation to the generative likelihood. The bottom-up classifier is trained to mimic the behavior of the top-down generative model. We describe our hybrid method in more detail in section 4.2.

3. The Jigsaw Model

In this section, we describe the probabilistic model used to learn a jigsaw from a set of training images [10]. The aim is to learn a jigsaw image, such that pieces of the jigsaw are similar in appearance to several regions of the training images and are as large as possible for a particular accuracy of reconstruction. These regions are allowed to be of arbitrary shape. In addition, the jigsaw is required to be exhaustive, so that the entirety of each training image can be reconstructed approximately using only pieces from the jigsaw image. Hence, the jigsaw captures repeated structures in the training image set. For instance, applied to a set of face images, the jigsaw captures both the appearance and the shape of eyes, noses and mouths [10].

A jigsaw $\mathbf{J}$ is defined as an image such that each pixel $\mathbf{z}$ in $\mathbf{J}$ has an intensity value $\mu(\mathbf{z})$ and an associated variance $\lambda^{-1}(\mathbf{z})$, so λ is the inverse variance, also called the precision. A jigsaw piece is a set of spatially grouped pixels in $\mathbf{J}$. We can combine many of these pieces to generate images, noting that pixels in the jigsaw be used in multiple image locations. For each image $\mathbf{I}$, we have an associated *offset map* $\mathbf{L}$ of the same size which determines the jigsaw pieces used to make that image. This offset map defines a position in the jigsaw for each pixel in the image, such that more than one image pixel can map to the same jigsaw pixel. Each entry in the offset map is a two-dimensional offset $\mathbf{l}_i = (l_x, l_y)$, which maps a 2D point i in the image to a 2D point $\mathbf{z}$ in the jigsaw using $\mathbf{z} = (i - \mathbf{l}_i) \bmod |\mathbf{J}|$, where $|\mathbf{J}| = (\text{width}, \text{height})$ are the dimensions of the jigsaw. Notice that if two adjacent pixels in the image have the same offset label, then they map to adjacent pixels in the jigsaw.

To explain an image using coherent pieces from the jigsaw, a Markov random field is defined on the offset map that encourages neighboring pixels to have the same offsets.

$$P(\mathbf{L}) = \frac{1}{Z} \exp\left[- \sum_{(i,j) \in E} \psi(\mathbf{l}_i, \mathbf{l}_j) \right] \tag{1}$$

where E is the set of edges in a 4-connected grid. The interaction potential ψ defines a Pott's model on the offsets, $\psi(\mathbf{l}_i, \mathbf{l}_j) = \gamma \, \delta(\mathbf{l}_i \neq \mathbf{l}_j)$, where γ is a parameter which influences the typical size of the learned jigsaw pieces. The choice of γ influences the granularity of segmentation of the image. For our experiments in this paper, we fixed γ at 6.

Given the offset map and the jigsaw, the probability distribution of each image is assumed to be independent for each pixel. Unlike [10], which used a Gaussian appearance model, we assume that the probability distribution for each image pixel is a mixture of a Gaussian and a uniform distribution (Fig. 1),

$$P(\mathbf{I} \,|\, \mathbf{J}, \mathbf{L}) = \prod_i \Big[\pi \mathcal{N}(I_i; \mu(i - \mathbf{l}_i), \lambda(i - \mathbf{l}_i)^{-1}) + (1 - \pi) \text{Uniform}(I_i) \Big] \tag{2}$$

where the product is over image pixel positions and both subtractions are modulo $|\mathbf{J}|$. The use of a mixture distribution has the effect of making the model more robust, and also allows for sparse inference methods to be used (see section 4.1). We fixed $\pi = 0.9$ in our experiments. For multi-channel images (*e.g*. RGB), separate mean and precision parameters used for each channel.

We place independent Normal-Gamma prior on μ and λ

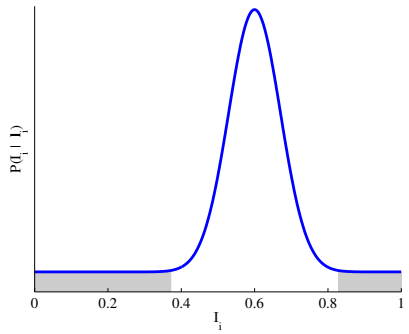


Figure 1. **Sparse structure of the likelihood function.** Plot of the likelihood $P(I_i|\mathbf{l}_i)$ for a particular offset $\mathbf{l}_i$. As the likelihood is a mixture of a Gaussian and a uniform, it is effectively constant for a range of values of I_i, shown shaded. Hence, it can be accurately represented by a sparse message, many of whose terms are identical.

for each jigsaw pixel $\mathbf{z}$:

$$P(\mathbf{J}) = \prod_{\mathbf{z}} \mathcal{N}(\mu(\mathbf{z}); \mu_0, (\beta\lambda(\mathbf{z}))^{-1}) \, \mathrm{Gamma}(\lambda(\mathbf{z}); a, b). \tag{3}$$

This prior ensures that the behavior of the model is well defined for unused regions. For our experiments, we fix the hyper parameters μ to 0.5, β to 1, b to three times the inverse data variance and a to the square of b.

4. Efficient Inference of the Offset Maps

The model defines the joint probability distribution on a jigsaw $\mathbf{J}$, a set of images $\mathbf{I}_1 \ldots \mathbf{I}_N$, and their offset maps $\mathbf{L}_1 \ldots \mathbf{L}_N$ to be

$$P\left(\mathbf{J}, \{\mathbf{I}_n, \mathbf{L}_n\}\right) = P(\mathbf{J}) \prod_{n=1}^{N} P(\mathbf{I}_n|\mathbf{J}, \mathbf{L}_n) P(\mathbf{L}). \tag{4}$$

In [10], an iterative approach is described for maximizing this joint probability that required alternately optimizing the offset maps $\{\mathbf{L}_n\}$ and the latent jigsaw image $\mathbf{J}$. The bottleneck in their procedure is the optimization of the offset maps, which used the alpha-expansion graph cut algorithm of [3]. This method scales roughly linearly with the number of pixels in the jigsaw and hence becomes prohibitively expensive for learning jigsaws of size greater than 100×100 pixels. To overcome this bottleneck, we propose using a variant of belief propagation (BP) which exploits the fact that many of the messages required during BP can be sparsely represented.

4.1. Sparse Belief Propagation

Optimizing the offset maps is a challenging problem due to the large state space of the offset for each pixel. A common approach to tackling this problem is to prune the state space of a variable by disallowing states for which there is little local support [4, 11]. However, this method is vulnerable to pruning out states incorrectly when the local evidence is insufficiently informative for accurate pruning. A promising alternative involves using a message-passing algorithm with sparsely represented messages, such that the true messages can be well approximated by their sparse counterparts. For example, Pal *et al.* [14] use a forward-backward algorithm and approximate each message $p(l)$ by a mixture of Kronecker delta functions $q(l)$ chosen to be within a fixed Kullback-Leibler divergence of the true message,

$$q(l) = \sum_{s \in S} q_s \delta(l = s). \tag{5}$$

In [14] it was shown that finding the approximate message with K delta functions that minimizes $\mathrm{KL}(q||p)$ requires simply retaining the largest K elements of p and renormalizing.

A problem with the above approach is that when messages are almost uniform, a very large number of delta functions is required to achieve a sufficiently good approximation, and so efficiency is lost. We overcome this problem by adding a uniform distribution to the mixture of delta functions, so that the sparse message has the form

$$q(l) = q_0 + \sum_{s \in S} q_s \delta(l = s). \tag{6}$$

Unfortunately, finding the sparse message that minimizes $\mathrm{KL}(q||p)$ does not now have a closed-form solution. Instead we retain the largest K elements of p and, rather than re-normalizing, evenly distribute the remaining probability mass amongst the remaining states (those not in S). To test the accuracy of this method, we generated random messages by sampling p vectors of size 1000 from a Dirichlet distribution. The 'peakiness' of the messages was varied by varying the Dirichlet pseudo-count parameters from 0.01 to 100. For $K = 100$, when using the approximation (5) the average KL divergence $\mathrm{KL}(q||p)$ was 1.1. However, when using our new approximation (6), the average KL divergence dropped to 0.3, indicating a much better approximation of the true message.

To optimize the jigsaw offset maps, we apply this approximation to max-product BP. We use sparse messages and beliefs throughout (unlike [14] which computes a dense belief and then finds the sparse equivalent). The messages are represented in log form and, since max-product BP is invariant to message normalization, each message is normalized so that $\log q_0$ is zero. The message $q(l)$ is then represented as a sorted list of the states in S and a corresponding list of the values $\log q_s$, requiring $O(K)$ memory.

The required message operations during Sparse BP are: multiplication of two messages, finding the pointwise maximum (upper envelope) of two messages and finding $q'(\mathbf{l}_j) = \max_{\mathbf{l}_i} q(\mathbf{l}_i)\psi(\mathbf{l}_i, \mathbf{l}_j)$. Since the messages are

sorted, all three of these operations can be achieved in $O(K)$ time. Interestingly, the latter operation actually leads to greater sparseness, since many of the entries of q' have the same value.

As our robust likelihood $P(I_i|\mathbf{J}, \mathbf{l}_i)$ can be represented to machine precision by a sparse message (Fig. 1), we can achieve the same results as belief propagation with full messages (full BP) whilst achieving significant memory and time savings (see section 4.2.1). However, if we desire additional improvements in efficiency, we can make the messages even more sparse, at the risk of leading to a poor approximation. Rather than following [14] and maximizing sparseness within some KL divergence, we instead incorporate longer range image information to minimize the risk of incorrectly pruning message states. This is achieved by exploiting bottom-up information in a hybrid approach.

4.2. Hybrid Belief Propagation

As we have seen, the likelihood function $P(I_i|\mathbf{J}, \mathbf{l}_i)$ defines a somewhat sparse message over the jigsaw locations $\mathbf{l}_i$ by taking into account the appearance of the single pixel i. For example, if pixel i is blue, jigsaw locations whose colors are dissimilar to blue will have approximately the same likelihood, given by the q_0 term in the sparse message. However, if we could take into account the appearance of image pixels around the ith pixel, we would be able to make the message significantly more sparse. For example, if all neighboring image pixels were red, we could effectively discard jigsaw pixel locations with non-red neighbors. Rather than construct a heuristic to achieve this, we use a classifier to *learn* the relationship between the image patch around a pixel and the jigsaw location that pixel gets mapped to. We will see, such a classifier is able to use features of the image patch around each pixel to achieve much more efficient inference, with minimal loss in accuracy.

We wish to train a classifier $\mathbf{T}$ to approximate the conditional probability $P(l_i|\mathbf{I})$ in our generative model. This is achieved by using a set of training images for which the corresponding offset maps $\mathbf{L}$ have already been computed. The classifier learns to predict the jigsaw location $\mathbf{l}_i$ for each pixel i of the training images given the surrounding patch of the image $\mathbf{I}$. Hence, it learns a (local) approximation to $P(l_i|\mathbf{I})$, which we will denote $\widetilde{P}(l_i|\mathbf{I}, \mathbf{T})$. The classifier $\mathbf{T}$ needs to be both efficient to train and to apply. Hence, we follow [13] and use a decision tree classifier trained with an entropy loss criterion values of pixels in particular relative locations to the pixel being classified. After learning the structure of the decision tree, we compute $\widetilde{P}(l_i|\mathbf{I}, \mathbf{T})$ by finding the histogram of jigsaw locations for those pixels which were assigned to each leaf node.

We use this bottom-up prediction to sparsify the generative likelihood by removing delta functions at locations with zero counts in $\widetilde{P}(l_i|\mathbf{I}, \mathbf{T})$ (see Figure 2). In other words,

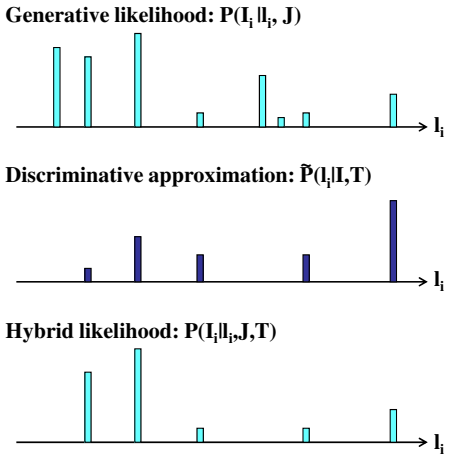


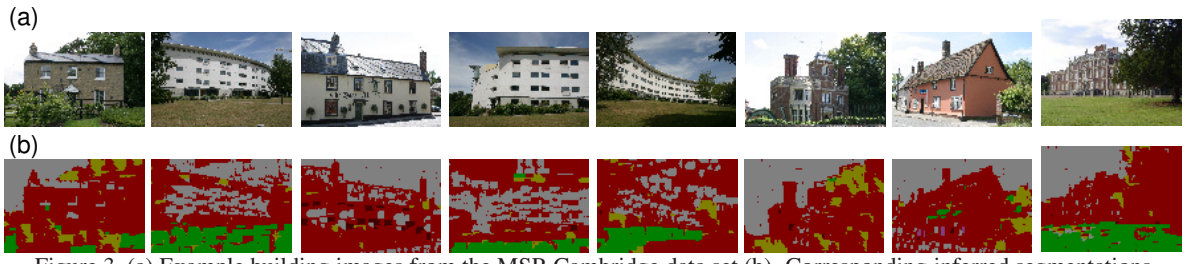
Figure 2. **Construction of the hybrid likelihood.** Top: the generative likelihood $P(I_i \mid \mathbf{l}_i, \mathbf{J})$ for a particular pixel. Middle: the discriminative probability $\widetilde{P}(\mathbf{l}_i \mid \mathbf{I}, \mathbf{T})$ for the offset l_i given the image and the decision tree. This is trained to approximate $P(\mathbf{l}_i \mid \mathbf{I})$ under the generative model. Bottom: Hybrid likelihood $P(I_i \mid \mathbf{l}_i, \mathbf{J}, \mathbf{T})$ which is equal to $P(I_i|\mathbf{l}_i, \mathbf{J})$ masked by the discriminative likelihood.

the discriminative prediction is used to mask the generative likelihood function. This method has the property that it gives the same solution as full BP on the training images. Hence, it is reasonable to expect that it will give good solutions on test images which have similar local image appearance. The exact quality of the approximation will depend on the generalization capabilities of the classifier.

4.2.1 Efficiency of Sparse BP and Hybrid BP

We are now interested in exploring the efficiency of Sparse BP and Hybrid BP when used to infer the offset maps for a pre-learned jigsaw. For this, we made use of 30 images of scenes containing buildings, taken from the Microsoft Research Cambridge object recognition data set [15]. We used 20 images for training, and kept 10 for testing. Figure 3a shows examples from the test set. We learned the 72×72 jigsaw shown in Figure 4a using the learning method of [10] from the 20 training images. The jigsaw size is relatively small as it is learned using existing method.

Given this jigsaw, we applied Hybrid BP with decision tree sizes from 1 to around 1800 nodes to learn the offset maps for the test images. Bear in mind that Hybrid BP is equivalent to Sparse BP when the tree size is 1. In Figure 4b we show the memory requirements of Hybrid BP as a percentage of the memory needed for full BP, for different decision tree sizes. Sparse BP is able to infer the offset maps using only 46% of the memory of a full BP implementation whilst guaranteeing the same solution. Hybrid BP is able to achieve almost arbitrary memory savings, however it is no longer guaranteed to find the same solution as full BP for



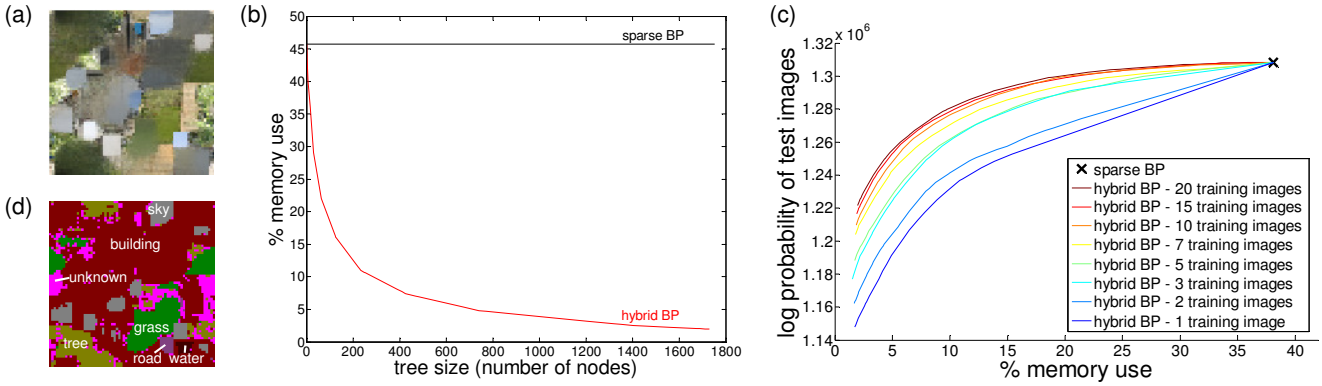


Figure 4. **(a)** The 72×72 jigsaw used to test Hybrid BP. **(b)** Memory usage of Sparse BP and of Hybrid BP for varying sizes of decision tree. Sparse BP can achieve equivalent to standard BP using only 46% of the memory. Hybrid BP can further reduce the memory requirements at the cost of a reduction in accuracy. **(c)** Accuracy of inference for Hybrid BP against memory use for different sizes of training set. High accuracy can be achieved on a set of test images with as little as $10 - 15\%$ of the memory needed for standard BP. **(d)** Class map showing the most likely object class at each jigsaw location.

test images. In general, as the decision tree is made larger to reduce memory use, the generalization performance reduces and so we expect lower accuracy on test images.

We also investigated the time requirements for applying 30 iterations of Hybrid BP. As expected, we found that the time varied linearly with the memory use, ranging from 36 seconds for 40%, down to two seconds for 2%. From now on, we will report results in terms of memory use, but it is safe to assume that similar percentage savings are made in terms of computation time.

We investigated this generalization performance by applying Hybrid BP to infer the offset maps for ten test images and using the joint probability (4) as a measure of the quality of this inference, since Hybrid BP should be maximizing this quantity. The joint probability is found normalized by an unknown but constant factor as the normalization term Z in $P(\mathbf{L})$ is intractable to compute. In Figure 4c, we present the joint probability of the test images as a function of the memory usage obtained by training decision trees using varying numbers of training images. Every plot in the figure is averaged across six experiments. This figure illustrates that for a fixed memory size, the generalization performance improves with training set size, but eventually saturates (in this case when around 10 training images are used). As expected, in all cases, when the memory used by Hybrid BP approaches that of Sparse BP, we recover the latter's performance.

Figure 4c shows that there is a trade-off between the accuracy (as measured by the joint probability) and the memory usage. We can significantly decrease the memory requirements if we are willing to accept the corresponding reduction in accuracy. The degree of accuracy required will depend on the application the jigsaw is being used for. In this paper, we select the accuracy of inference by using the task of segmenting an image into regions corresponding to the different object classes present in the image.

To use the jigsaw for this object recognition task, we first need a *class map*: a distribution over the class label for each jigsaw pixel. We learn this map from training images which are partially labeled with ground truth pixel-wise labels of the following eight classes: sky, building, water, people, road, mountain, grass and tree. Note that, in this small training set, there are only two or three examples of several of these classes. The resultant class map is shown in Fig. 4d. To label a test image, we first infer the offset map for that image, then assign to each pixel the most probable class of the corresponding jigsaw location.

For each resulting set of offset maps inferred during the above experiments, we compute the recognition accuracy on the test set, defined as the percentage of correctly classi-

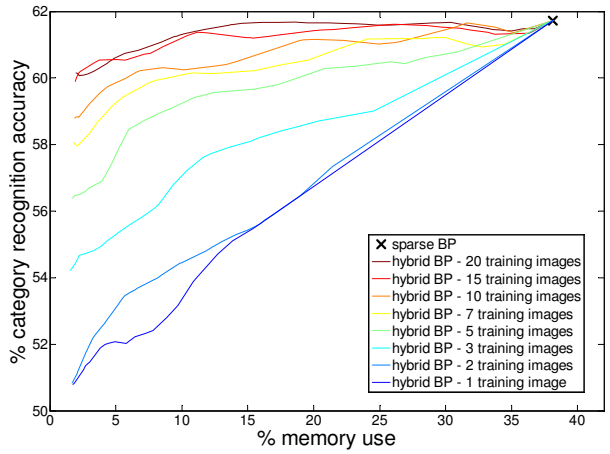


Figure 5. **Recognition accuracy against memory use when applying Hybrid BP with different sizes of training set for learning the tree.** The jigsaw was pre-learned on all 20 images and is the same in all cases. The number of training images refers to the size of the image set used to train the bottom-up decision tree. When all 20 training images are used, high recognition accuracy can be achieved using just $10-15\%$ of the memory of standard BP. The curves converge to the solution for Sparse BP (the cross) since, as the tree gets shallower and more memory is used, Hybrid BP becomes equivalent to Sparse BP.

fied pixels across all test images. The results are shown in Fig. 5, again averaged over six runs. Note that the jigsaw and class map remain fixed throughout – only the mapping from the test images to the jigsaw changes.

When trained on all 20 training images, there is negligible loss in recognition accuracy compared to full BP until the memory requirements are reduced to around 15%. When the offset maps are inferred using even less memory, the accuracy slowly drops off, reducing by around $1-2\%$ as the memory requirements reduce to less than 5% of full BP. As expected, the recognition accuracy is reduced when the decision tree is trained with fewer images, due to poorer generalization performance.

Consider learning a very tiny jigsaw. The jigsaw pixels end up corresponding to the different colors found in the training images and the inferred patches are individual pixels. In this case, the surrounding pixels would have no effect on where a particular pixel maps to in the jigsaw and so Hybrid BP does no better than Sparse BP. Conversely, for larger jigsaws, the inferred patches are larger and Hybrid BP has an increasing efficiency advantage. The improved relative efficiency of Hybrid BP with increasing jigsaw size is shown in Figure 6 for the region where the size<96. In this region the tree depth was varied to maintain the accuracy of learning relative to full BP. As can be seen, the percentage of memory use drops with increasing jigsaw size. For jigsaws of size 96×96 and larger, we could not apply full BP due to memory constraints and so the tree depth was selected so as to keep within the available memory.

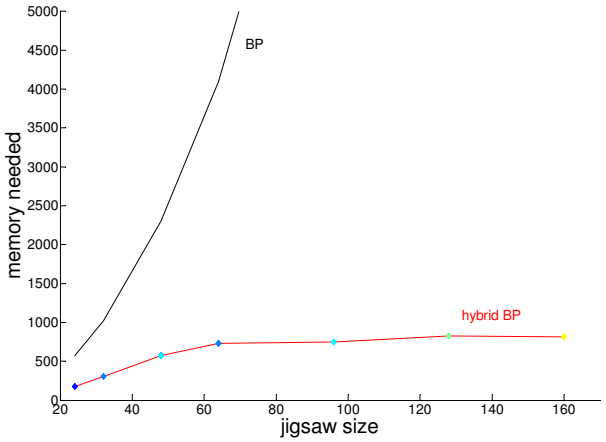


Figure 6. **Memory needed for different jigsaw sizes.** The plot shows the memory needed for a BP message when learning different sizes of jigsaw, where the size is the length of side of a square jigsaw. The memory needed for full BP therefore grows with the square of the jigsaw size. As we have constrained memory (1000), we restrict Hybrid BP to use less than this amount at the cost of a reduction in the accuracy of inference.

5. Hybrid Jigsaw Learning

We will now show how Hybrid BP can be used to *learn* a jigsaw from a set of training images. As mentioned in section 4, learning is accomplished in [10] by alternately maximizing the joint probability (4) with respect to the offset maps $\{\mathbf{L}_n\}$ and the latent jigsaw image $\mathbf{J}$. Figure 7a gives an overview of this learning method. The jigsaw is initialized by setting the mean to random pixel values from the training images and setting the variance to the expected value under the prior.

We now wish to modify this learning procedure to use Hybrid BP when updating the offset maps. This requires that a decision tree has been trained on a set of images and their corresponding offset maps. Unfortunately, during the initialization of the algorithm, we do not yet have offset maps for the training images and so we cannot perform supervised learning of the decision tree. Instead, we learn the tree in an unsupervised manner so as to cluster the training image pixels into roughly equally sized clusters. Rather than using an entropy criterion for tree learning, we instead use a criterion that favors splitting the training data at each tree node into similar amounts across the two child nodes. This leads to a balanced tree which have the property that pixels assigned to a particular leaf have surrounding image patches of similar appearance. In other words, the tree effectively clusters image patches. After learning the tree, for each leaf, we define $\widetilde{P}(l_i|\mathbf{I},\mathbf{T})$ to be uniform over a randomly selected proportion (50%) of the memory allocated for sparse BP.

Given this initialization of the decision tree classifier, we now apply Hybrid BP to infer the offset maps. From then

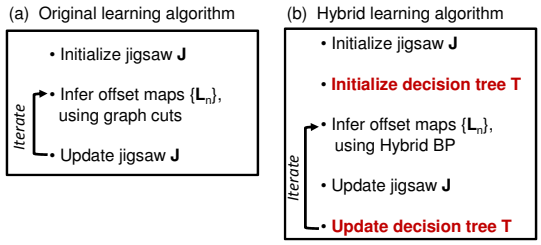


Figure 7. **Comparison of algorithms for learning jigsaws.** (a) Original learning algorithm of [10] (b) Hybrid algorithm using Hybrid BP, with new steps shown in red (see text for details).

on, the tree can be updated as described in section 4.2. The overall hybrid learning algorithm is shown in Figure 7b. To speed-up learning the tree, we do not expand a leaf whose histogram has reached a selected sparsity threshold (typically, 65% of the memory allocated for BP). This criterion attempts to achieve maximum generalization by making the messages sparse enough to fit into the allocated memory, but no more. In addition, instead of learning the structure of the tree at each iteration, we first recompute the leaf histograms for the current tree using the new offset maps, and learn the new tree only when the memory requirements of any leaf goes above the allocated amount. Avoiding updating the tree structure in this way dramatically speeds up learning. We also add 5% randomly chosen jigsaw locations to $\widetilde{P}(l_i|\mathbf{I},\mathbf{T})$ to encourage previously unused jigsaw locations to become used.

5.1. Analysis of hybrid learning

To test the effectiveness of our learning scheme, we compared the performance of hybrid jigsaw learning with the original 'generative' jigsaw learning method of [10]. We again made use of the building images described in section 4.2.1, using the same 20 images as the training set. We learned 72×72 jigsaws using each of the two approaches. This jigsaw size was selected because it is the largest jigsaw that we can learn using the generative learning method before running into memory limitations. We can control the memory requirements of the hybrid learning by varying the depth of the learned decision trees and so are able to learn much larger jigsaws.

We perform hybrid jigsaw learning for a range of tree depths by varying the convergence criterion of the decision tree learning. Since the decision trees are being updated during learning, their size will change over time. For these experiments, the trees ranged in size up to 730 nodes at the first iteration, reducing to 345 nodes at convergence. To measure the quality of the learned jigsaws, we again use the joint probability (see section 4.2.1) which we aim to maximize. The results are shown in Fig. 8, where we plot the log joint probability and the recognition accuracy against the memory use during the final iteration of learning (this

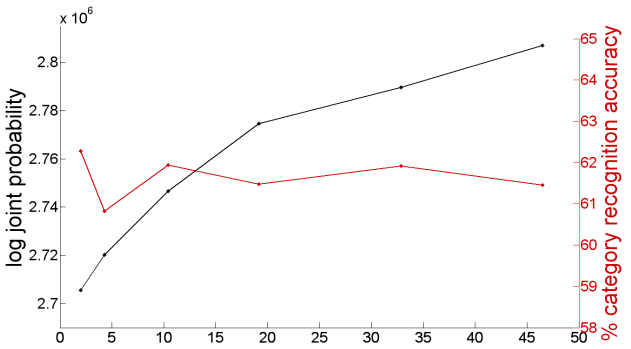


Figure 8. **Accuracy of the hybrid learning against memory use** Learning accuracy remains high even at 20% memory use and becomes equivalent to generative learning at 45% memory use. The recognition accuracy is relatively stable within a 1% variation

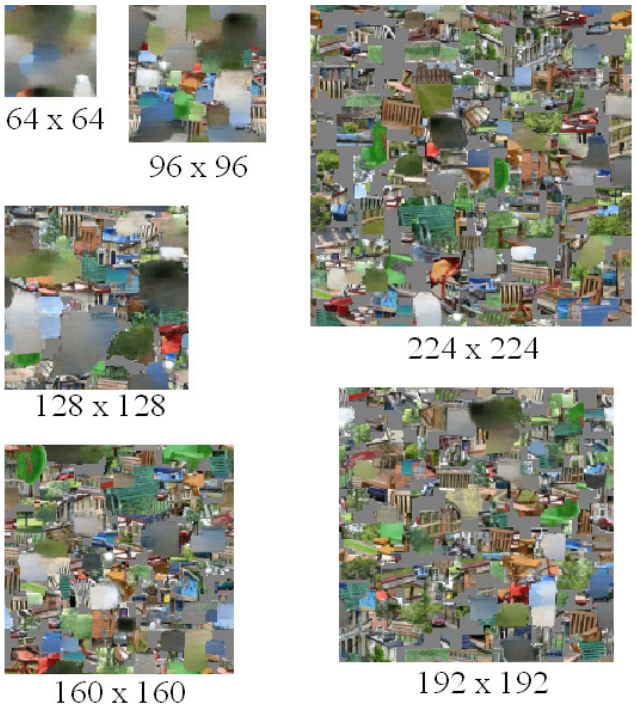


Figure 9. **Jigsaws of various sizes learned from 100 images.** Larger jigsaws capture larger and more complex image features, until they eventually over-fit to particular training image regions.

is representative of the memory use overall). The results show a fall-off in log joint probability quantities for reduced memory use with a similar form to that of Hybrid BP. Thus, we can reduce memory usage for learning jigsaws, if we are willing to accept a small reduction in accuracy.

5.2. Object recognition/segmentation

We applied our hybrid jigsaw learning method to the task of object recognition and segmentation, to explore the effect of jigsaw size on recognition performance.

For this task, we made use of the training set from sec-

tion 4.2.1 and also a larger image set of 100 images from the Microsoft Cambridge data set, containing 14 different classes: sky, building, water, people, road, mountain, grass, tree, chair, car, boat, sign, bicycle and bird. Jigsaws were learned with sizes ranging from 32×32 to 160×160 for the small training set and from 64×64 to 224×224 for the larger data set. Figure 9 shows the jigsaws learned using the larger dataset. While smaller jigsaws capture essentially colour information, larger jigsaws capture larger image structures, such as bench pieces, textures and so on.

Each learned jigsaws was applied to the problem of object recognition/segmentation. For each dataset, we randomly selected 75% of the images (75 and 25 images for larger and smaller datasets, respectively). We used the mapping of these selected images into the jigsaw and their ground truth labels to construct a class map, as described in section 4.2.1. Using this class map, we inferred class labels for each pixel of the remaining images in the corresponding data set. Both experiments were repeated for 20 different random splits. Figure 10 shows, for both experiments, the recognition accuracy (averaged across the random splits), as a function of jigsaw size.

We are able to recognise objects in unlabelled images because structures in the jigsaw are shared between the labelled and unlabelled images, and these labels are transferred across. As the jigsaw size increases, the image structures become larger and more informative as to the object class, and so accuracy improves. Eventually, the jigsaw becomes so large that some jigsaw regions are only found in one image, and so are not shared between the labelled and unlabelled images. In this case, the jigsaw has to over-fit to particular image regions. The size of jigsaw that gives peak performance has both large shared structures and a high degree of sharing of these structures. Hence, this optimal size is a function of the training set size. We demonstrate this in Figure 10, where, we find the peak in accuracy is at a larger jigsaw size for the larger data set. The drop in accuracy in the larger dataset is due to the larger number of categories present (14 instead of 8) in the training set.

6. Conclusions

In this paper, we presented a novel hybrid method for performing inference and learning in the jigsaw model, that allows significant reduction of memory usage and computation time, for minimal loss in accuracy. To demonstrate the effectiveness of our method, we presented results applying large jigsaws to a small object recognition task.

Being able to learn large jigsaws efficiently will allow this model to be used for tasks such a motion segmentation, object recognition with large data-sets and wide-baseline matching. We propose to investigate the performance of this model for these varied tasks. In particular, an advantage of this model is that it is learned unsupervised and hence will discover object parts without the need for labelling. We propose to investigate how the performance improves with the addition of large amounts of unlabelled image data for each of these applications.

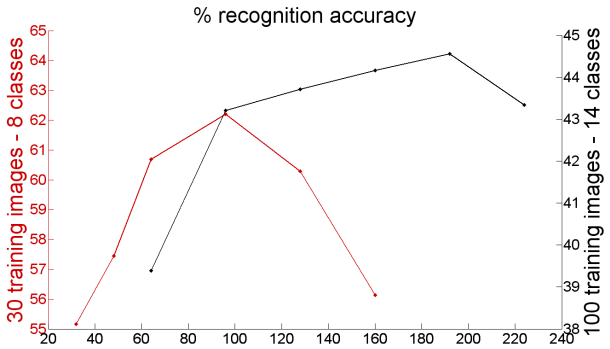


Figure 10. **Recognition accuracy against jigsaw size.** Accuracy increases with size as larger and more informative image structures are learned, until these structures become too large and overfit to particular training images. Note that red curve corresponds to various sized jigsaws trained using 30 images consisting of 8 classes, while the black curve corresponds to jigsaws trained using 100 images with 14 classes in total.

References

[1] E. Borenstein, E. Sharon, and S. Ullman. Combining top-down and bottom-up segmentation. In *Proceedings IEEE workshop on Perceptual Organization in Computer Vision, CVPR 2004*, 2004.

[2] G. Bouchard and B. Triggs. The trade-off between generative and discriminative classifiers. In *IASC 16th International symposium on computational statistics*, pages 721–728, 2004.

[3] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11), 2001.

[4] J. Coughlan and S. Ferreira. Finding deformable shapes using loopy belief propagation. In *ECCV*, pages 453–468, 2002.

[5] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, volume 2, pages 264–271, June 2003.

[6] G. Hinton and V. Nair. Inferring motor programs from images of handwritten digits. In *NIPS 18*, pages 515–522. MIT Press, Cambridge, MA, 2006.

[7] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[8] D. Huttenlocher, D. Crandall, and P. Felzenszwalb. Spatial priors for part-based recognition using statistical models. In *Proceedings of IEEE CVPR*, 2005.

[9] N. Jojic, B. Frey, and A. Kannan. Epitomic analysis of appearance and shape. In *ICCV*, 2003.

[10] A. Kannan, J. Winn, and C. Rother. Clustering appearance and shape by learning jigsaws. In *NIPS 19*, Cambridge, MA, 2007. MIT Press.

[11] N. Komodakis and G. Tziritas. Image completion using global optimization. In *IEEE CVPR*, 2006.

[12] J. Lasserre, C. Bishop, and T. Minka. Principled hybrids of generative and discriminative models. In *IEEE CVPR*, 2006.

[13] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *CVPR05*, pages II: 775–781, 2005.

[14] C. Pal, C. Sutton, and A. McCallum. Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2006.

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