Consistent Image Analogies using Semi-supervised Learning

Li Cheng, S.V. N. Vishwanathan, Xinhua Zhang
NICTA and Australian National University
Canberra, Australia
li.cheng,SVN.Vishwanathan,Xinhua.Zhang@nicta.com.au

Abstract

In this paper we study the following problem: given two source images A and A', and a target image B, can we learn to synthesize a new image B' which relates to B in the same way that A' relates to A? We propose an algorithm which a) uses a semi-supervised component to exploit the fact that the target image B is available apriori, b) uses inference on a Markov Random Field (MRF) to ensure global consistency, and c) uses image quilting to ensure local consistency. Our algorithm can also deal with the case when A is only partially labeled, that is, only small parts of A' are available for training. Empirical evaluation shows that our algorithm consistently produces visually pleasing results, outperforming the state of the art.

1. Introduction

Consider the following situation: during your vacation you shoot pictures of several beautiful spots, some of which turn out to be blurred. Standard tools like Photoshop provide filters to sharpen such images. Unfortunately, these filters are far from perfect: the resulting images need to be hand *touched* to achieve a visually pleasing outcome. This process is tedious, error prone, and time consuming, especially when many similar photographs have to be deblurred. In order to automate this type of processes, the image analogies paradigm was proposed by [5]. The basic idea is simple yet elegant: The algorithm takes as input two images A, A' (the unfiltered and filtered source images respectively). Now, given a new image B, it synthesizes a new filtered target image B' which relates to B in the same way that A' relates to A. For instance, if A' is a deblurred version of A, then B' is a deblurred version of B. The algorithm in [5] can be described as follows: each pixel in image B is assigned the label (*i.e.* the corresponding pixel value in image A') of a *similar* pixel in A. Similar pixels are found using a

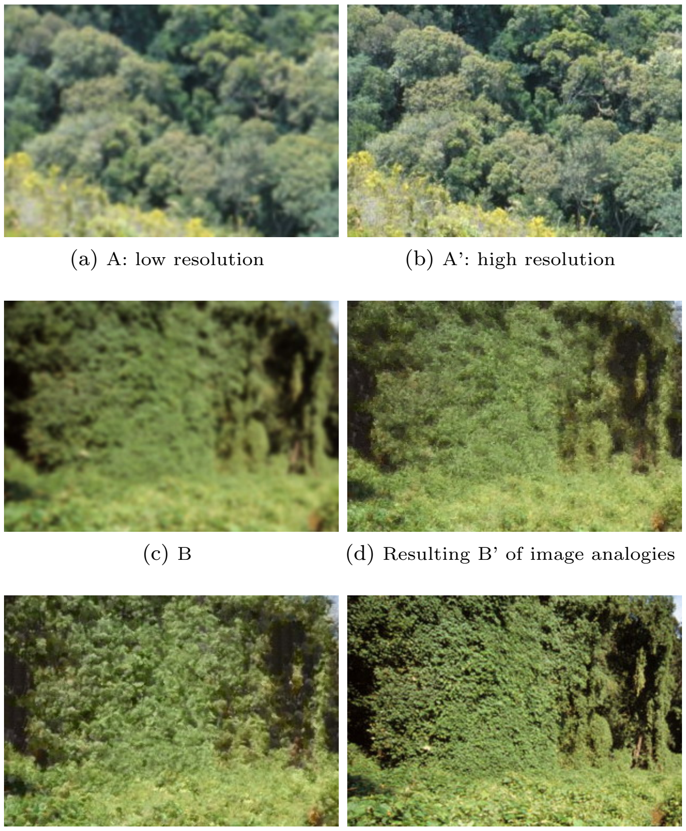

(a) A: low resolution
(b) A': high resolution
(c) B
(d) Resulting B' of image analogies
(e) Resulting B' of our algorithm
(f) Ground truth

Figure 1: An image super resolution example. Patch A, and its corresponding high resolution counterpart A' are sampled from a larger image. Given the large image and a few such high resolution parts, our algorithm transforms target patch B. The output of our algorithm is contrasted with that of image analogies, which had access to the large image as well as its *entire* high resolution counterpart.

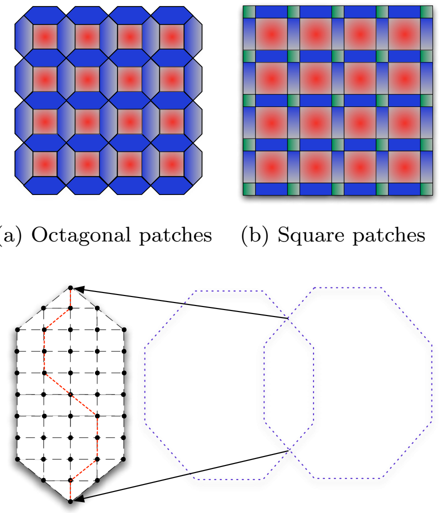
Euclidean metric search as well as a coherence search both of which employ an approximate nearest neighbor algorithm. Furthermore, pixels are coarsened at various levels, and this assignment is carried out at each level to ensure consistency.

As demonstrated by [5], the image analogies paradigm is rather successful in learning visually pleasing transformations on a wide range of tasks including image deblurring, super resolution, and style transfer. Unfortunately, this elegant paradigm also suffers from a few drawbacks. First, it requires the user to provide an exemplar image and a fully filtered output. Obtaining a fully filtered image might be very expensive. Second, since the algorithm uses a simple nearest neighbor query to label each pixel, there are marked boundary effects and the resultant images are generally *noisy* and ineffective to preserve texture structure under larger context (see Figure 1). This problem is partially mitigated by a multi-scale query and a coherence search to enforce local consistency among neighboring patches, but a principled solution is lacking. Finally, the algorithm does not exploit the fact that the unfiltered target image B (such as Figure 1c) is available at training time, and hence can be used as a source of unlabeled examples.

To address the issue of local consistency, especially around the boundaries, [2] proposed the image quilting algorithm. Given a texture A and a target image B, the algorithm divides B into small overlapping patches and produces candidate textures for these patches by randomly sampling from A. These candidate patches are then stitched together by a dynamic program which minimizes the error boundary. The image quilting algorithm produces impressive results for texture synthesis and transfer. However, since the emphasis is on local consistency, the internal structure of the whole image is ignored.

Freeman *et al.* [3, 4] propose a method that treats image inference as a Markov Random Field (MRF) problem, using multiple candidate patches found with nearest neighbor. It considers global consistency and is employed in applications such as super-resolution where abundant training images are available, however, it does not suit well to scenarios where it is expensive to obtain labels. By exploiting unlabeled examples, our algorithm gives consistently better results as shown in the experiments section. In the passing, we note that Rosales *et al.* [8] discuss an unsupervised method of this problem.

The aim of this paper is to address the problem of learning locally and globally consistent image filters from patches in a principled and unified manner. First, we exploit the fact that the target image B is available apriori, to warp the metric used to compare patches. Second, we encode local consistency conditions as edge potentials and propagate them across the label state space of a MRF to produce globally consistent assignments. Finally, we employ the image quilting algorithm to ensure that our assignments are locally consistent and visually pleasing. Figure 1 presents an illustrative example on image super resolution where our algorithm produces visually more appealing result (e) than that of image analogies (d), although both can be very different from the ground truth.


(a) Octagonal patches (b) Square patches

(c) Stitching two neighboring patches together

Figure 2: Our algorithm divides an input image into overlapping octagons as illustrated in (a). Each pixel either belongs to exactly one octagonal patch (these pixels are shaded red) or two octagonal patches (shaded blue). In contrast, if the image is divided into square patches as shown in (b), then each pixel may belong to one square patch (shaded red), two square patches (shaded blue), or four square patches (shaded green). (c) demonstrates the minimum error boundary when stitching two neighboring patches together by dynamic programming (see text for details).

2. Our Algorithm

As in the image analogies paradigm [5], we assume that we are given three images: A the unfiltered source, A' the filtered source, and B the unfiltered target. The aim is to produce the filtered target B'. As opposed to the image analogies setting, which requires the user to provide a fully labeled image, A', our algorithm also allows the following more user-friendly scenarios: a) Given an image and some *partially* revealed labels, label the rest of the image, and b) Given a source image and partially revealed labels, label other images with similar content. In fact, as will be explained in

Section 2.2.1, our algorithm exploits both labeled and unlabeled examples to modify the metric used for the nearest neighbor search. This enables us to produce visually pleasing results with very limited input. We will illustrate various steps with a running example shown in Figure 3. The resultant superresolved B' of the comparison algorithms are presented in panels (c)–(f), where a portion of these B' corresponding to the blue rectangle region of B are displayed here.

2.1. Dividing the Image into Patches

We divide all three images into overlapping octagonal patches as illustrated in Figure 2. Let $\mathcal{A} = \{x_i\}_{i=1}^{m}$, $\mathcal{B} = \{x_j\}_{j=m+1}^{n}$, and $\mathcal{A}' = \{y_i\}_{i=1}^{m}$ denote the patches in images A, B, and A' respectively, and denote $\mathcal{X} = \mathcal{A} \cup \mathcal{B}$. In this case, we are given m labeled observations $\{x_i, y_i\}$ and $n - m$ unlabeled observations $\{x_j\}$. On the other hand, if only $d < m$ patches of image A' are revealed to us, that is, the algorithm has access to only d labels, then we assume without loss of generality that $\{x_i, y_i\}_{i=1}^{d}$ are the d labeled observations and $\{x_j\}_{j=d+1}^{n}$ are the $n - d$ unlabeled observations.

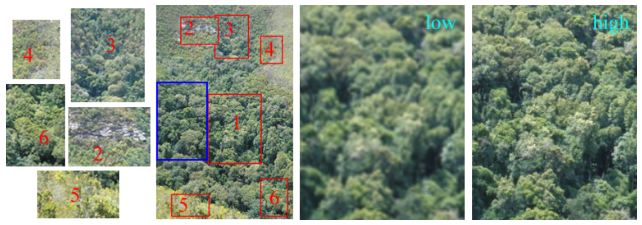
Each patch x in $\mathcal{X}$ is represented by a feature vector $\phi(x)$. Following standard practice, we assume that the feature space is a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$, whose kernel $k(x, x') := \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$ is readily available; for our experiments we used local texture features and a Gaussian kernel (see Section 3).

2.2. Nearest Neighbor Query

Next we perform a nearest neighbor query. A naive implementation simply uses the metric induced by the kernel $k(x, x')$, and labels each patch in $\mathcal{B}$ by the label of its nearest neighbor in $\mathcal{A}$. Unfortunately, this leads to a visually unsatisfactory result as shown in Figure 3e. Instead, our algorithm exploits the fact that besides labeled patches in $\mathcal{A}$, unlabeled patches in $\mathcal{A}$ and $\mathcal{B}$ are also available, and hence the metric can be modified appropriately via semi-supervised learning (SSL).

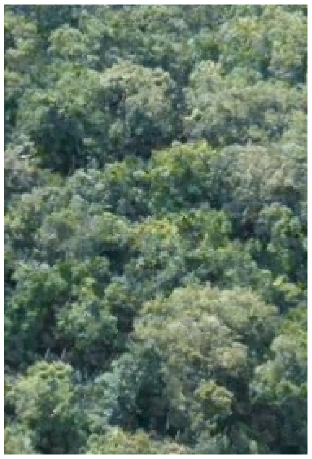
2.2.1 Modifying the Metric via Semi-Supervised Learning

Semi-supervised learning refers to the problem of learning from both labeled and unlabeled data. It has attracted considerable attention in recent years (see [11] for a comprehensive survey). We work with the graph based methods of [1] which construct a *feature graph*, $\mathcal{G}(\mathcal{V}, \mathcal{E})$. Each node of the feature graph represents an observation x (including both labeled and unlabeled) in feature space $\phi(x)$ and edges encode nearest neighbor relationships, *i.e.* $(i, j) \in \mathcal{E}$ iff j is among the k-nearest neighbor of i or vice versa.

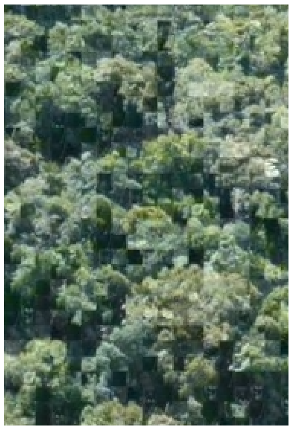


(a) A, A' and B

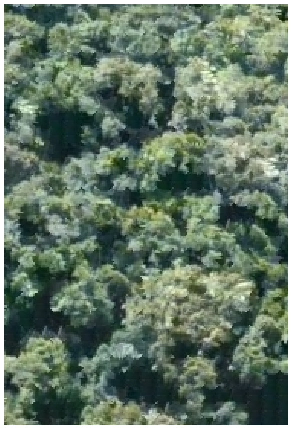
(b) Ground truth



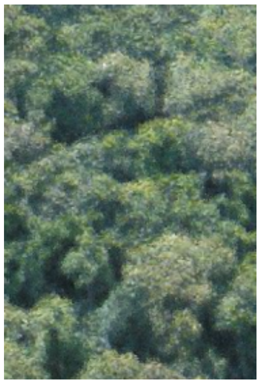
(c) B' of our algorithm



(d) B' of only inference



(e) B' of only inference and quilting



(f) B' of image analogies

Figure 3: Another super resolution example. Given the input image A and a few training parts A' the output of our algorithm is contrasted with those of inference, quilting and image analogies

The edges are often weighted to quantify the similarity between neighboring observations, *e.g.* $w_{ij} = \exp(-||\phi(x_i) - \phi(x_j)||^2/\sigma_w^2)$ which is used in our experiment. In supervised learning, the following regularized risk is minimized:

$$J(f) = ||f||^2_{\mathcal{H}} + \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i, f), \tag{1}$$

where $||f||^2_{\mathcal{H}}$ is the RKHS norm of f in $\mathcal{H}$ and $l(\cdot,\cdot,\cdot)$ is a loss function. Semi-supervised learning, in addition, also prefers functions with high smoothness on the graph, by minimizing the following data dependent regularized risk[1] over $f \in \mathcal{H}$:

$$J(f) = ||f||^2_{\mathcal{H}} + ||f||^2_{\mathcal{G}} + \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i, f), \tag{2}$$

where $|f||^2_{\mathcal{G}} := \boldsymbol{f}^\top L \boldsymbol{f}$, $\boldsymbol{f}$ denotes the vector $[f(x_1), \ldots, f(x_m), \ldots, f(x_n)]^\top$ and L denotes the Laplacian of $\mathcal{G}$. $L := D - W$, where W is the weight matrix with $W_{ij} = w_{ij}$ and D is a diagonal matrix with $D_{ii} = \sum_j w_{ij}$. Of particular interest to us is the fact that the semi-supervised learning algorithms supplement the regularizer $||f||^2_{\mathcal{H}}$ with a data dependent regularizer $||f||^2_{\mathcal{G}}$. Therefore, it is natural to ask whether there exists an RKHS $\tilde{\mathcal{H}}$ which satisfies: a) $||f||^2_{\tilde{\mathcal{H}}} = ||f||^2_{\mathcal{H}} + ||f||^2_{\mathcal{G}}$; b) its corresponding kernel $\tilde{k}$ can be computed explicitly and efficiently. This question was answered affirmatively by [9]. Letting $\mathbf{k}_x := [k(x_1, x), \ldots, k(x_m, x), \ldots, k(x_n, x)]^\top$ for any $x \in \mathcal{X}$, [9] showed that

$$\tilde{k}(x, x') := k(x, x') - \mathbf{k}_x^\top (I + LK)^{-1} L \mathbf{k}_{x'}, \tag{3}$$

and its associated RKHS, $\tilde{\mathcal{H}}$, satisfies our requirements. The appendix (Section 6) provides a simple proof of (3). The metric defined by this kernel, which is warped from the original RKHS $\mathcal{H}$ by using unlabeled data, reflects the intrinsic manifold on which $\mathcal{X}$ lies. Labeling each patch in $\mathcal{B}$ by the label of its nearest neighbor in $\mathcal{A}$ under this new metric results in a visually more pleasing result. For instance, several noticeable artifacts in Figure 3e disappear when using the modified metric as shown in Figure 3c.

2.3. Inference on a Markov Random Field

A nearest neighbor query does not preserve visual consistency with respect to the output image, and a pixel-level query even worsens the result as it also tends to ignore the higher-order image context. To overcome this problem, [5] coarsen the image at various levels, and perform a nearest neighbor query at each level to ensure consistency. This often results in noisy target images, as seen *e.g.* in Figures 1d and 3f, which are noticeably different from the target style.

Similar to [3, 4], we cast the problem of global consistency as an inference problem on a Markov Random Field (MRF). Our scheme works as follows: For each patch x_i in the target image B, we use the (modified) metric to find its k nearest neighbors in $\mathcal{A}$, denoted by $\{x_{i,1}, \ldots, x_{i,k}\}$, and labeled as $\{y_{i,1}, \ldots, y_{i,k}\}$. Next, we define discrete random variables $Z_i \in \{1, \ldots, k\}$. Every configuration of $\mathcal{Z} := \{Z_i\}$ defines a labeling for the image B by assigning the label $y_{i,z}$ to the patch x_i if Z_i takes on a value z. This formulation can also be extended to the case when each $Z_i \in \{1, \ldots, k_i\}$, that is, for each patch x_i we find k_i nearest neighbors in $\mathcal{A}$. For notational convenience we simply assume that $k_i = k$ for all i.

Recall that MRFs are undirected graphical models [6]. An undirected graphical model is defined by a graph[2], $G(V, E)$, whose node set V is in one-to-one correspondence with a set of random variables, and each edge in E represents a *dependency* between the corresponding random variables. We restrict our attention to MRFs over discrete random variables Z_i. There exists an edge between Z_i and Z_j if, and only if, patches x_i and x_j overlaps in the target image B, *i.e.* $x_i \cap x_j$ is a hexagon as in Figure 2c. By the Hammersley-Clifford theorem, the joint distribution of $\mathcal{Z}$ can be written as

$$p(\mathcal{Z}) \propto \prod_{i \in V} \psi_i(z_i) \prod_{(i,j) \in E} \psi_{ij}(z_i, z_j), \tag{4}$$

where ψ_i and ψ_{ij} are compatibility functions or potential functions on nodes and edges respectively. Our node compatibility function $\psi_i(z)$ takes into account the modified metric between patch $x_i \in \mathcal{B}$ and $x_{i,z} \in \mathcal{A}$. It is defined as $\psi_i(z) = c_1 \tilde{k}(x_i, x_{i,z})$. The edge compatibility function measures the compatibility between label assignments of two adjacent nodes and is defined as $\psi_{ij}(z, z') = c_2 \exp(-err_{ss}(y_{i,z}, y_{j,z'})/\sigma_e^2)$, where $err_{ss}(y_{i,z}, y_{j,z'}) \in \mathbb{R}$ is the sum of squared differences of the overlapping region between the two octagon patch labels $y_{i,z}$ and $y_{j,z'}$. c_1, c_2, and σ_e are positive tunable parameters. Loopy belief propagation [6] is employed to perform inference on this MRF, and the MAP configuration results in a globally consistent labeling of B.

While similar in vein to [3, 4], the inference method we used are different in several aspects: Instead of the

[1] Conventionally a term trading off the two regularizers is used, which we ignore for ease of exposition.

[2] This graph is distinct from the feature graph we discussed in the previous section.

complex multi-resolution representation of label field, our experiments show that it is visually indistinguishable for our algorithm to work on a single label resolution which removes the need to negotiate labels among various layers in the decoding phase; Secondly, as the training labels are sparse we also consider unlabeled examples and by employing a data dependent metric we capture smoothness in the feature space. Furthermore, to deal with overlapping regions, rather than taking the average ([4]) or simply cutting a straight line in between (*e.g.* Figure 3d), we adopted a technique detailed next.

2.4. Image Quilting

Recall that the target image B is divided into overlapping octagonal patches, which are labeled with patches from A'. As a result, the predicted B' consists of overlapping patches. Let s and t denote the source and sink vertices of a hexagonal overlap area, as depicted in Figure 2c. The naive approaches of averaging or cutting an overlapping patch by a straight line joining s and t are visually unsatisfactory. Here, we use the simple but effective image quilting algorithm due to [2]. This ensures local consistency, as shown in Figure 3e, where the blocky effect is considerably reduced as compared to Figure 3e. Similar effects can be observed in Figure 6e and 6f.

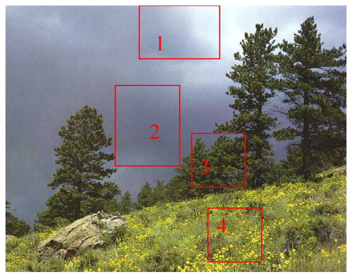
Formally, let y and y' be the labels of two neighboring patches. For each location (i, j) in the overlapping area, we associate a mismatch cost $e_{i,j} := \exp(-||p_{i,j} - p'_{i,j}||^2)$, where $p_{i,j}$ (resp. $p'_{i,j}$) denotes the labeling of the pixel due to y (resp. y') and $||p_{i,j} - p'_{i,j}||$ quantifies the discrepancy in label values. In Figure 2c, let s be the source, and t be the sink. To ensure local consistency, [2] propose to cut the overlap region by a path p from the source to the sink such that the total cost of the path is minimized. This problem can be solved efficiently by the following dynamic programming:

$$E_{i,j} = e_{i,j} + \min(E_{i-1,j-1}, E_{i-1,j}, E_{i-1,j+1}). \qquad (5)$$

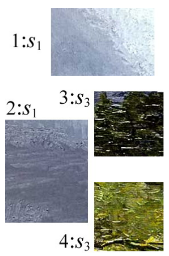
Reading off the entry corresponding to the sink vertex gives the minimum cost, while the optimal path (cut) can be found by backtracking. We employ quilting as the final post-processing step of our algorithm in order to ensure that the output is visually pleasing (see *e.g.* Figures 3c and as contrast, see 3d when without quilting).

3. Implementation Details

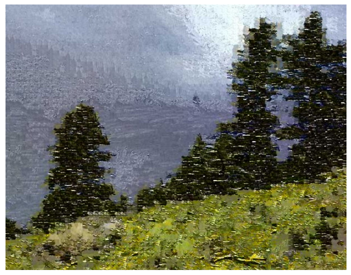
Before presenting our results, we describe some implementation details, including standard techniques of

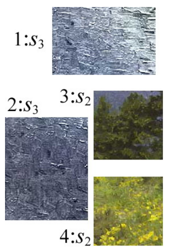


(a) A

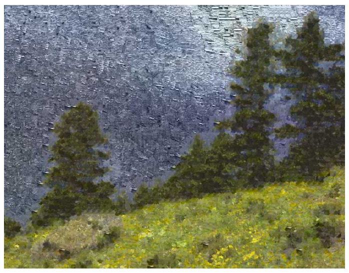


(b) A' using styles $s1$ and $s3$


(c) B' by our algorithm with (e) as A'



(d) A' using styles $s3$ and $s2$


(e) B' by our algorithm with (d) as A'

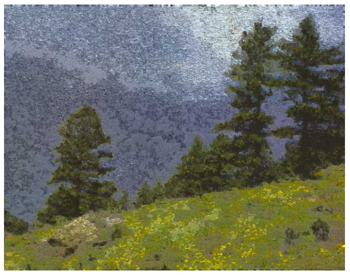

(f) B' by image analogies with (d) as A'

Figure 4: A style transfer example. Our algorithm can learn how to apply different styles to proper places of the image.

image manipulation, which we employ. Following [7], we work in the YUV space, where Y is the intensity (luminance) channel, and U, V are the chrominance channels that encode color. As noted in section 2, our algorithm uses octagonal patches instead of pixels as basic image elements. For all our experiments we fix the size of each octagonal patch to be 15 pixels in diameter, and set the width of the overlap region on each side to be 5 pixels.

The feature vector of a patch is composed of the pixels' intensity value, as well as its intensity gradient value after applying a variant of canny edge detector. Our experiments show that this simple feature vector representation, coupled with a Gaussian kernel whose variance is tuned for each experiment separately, is sufficiently rich to capture the local texture context of each octagonal patch.

The feature graph, $\mathcal{G}(\mathcal{V}, \mathcal{E})$, used for semi-supervised learning (see Section 2.2.1) is a t-nearest neighbor graph (with t chosen to be between 5 to 10 depending on the size of the input images). The σ_w used for defining edge weights w_{ij} is set to 1.

If A contains large homogeneous regions, then all the k neighbors of a patch x_i from the target image B might be very similar. To add "variety" to the candidate label set we randomly sample k candidates from the ck nearest neighbors of x_i. For our experiments, we fix $c = 3$ and $k = 5$. Finally, for the MRF we choose the tuning parameters $\sigma_e = 0.3$ and $c_1 = c_2 = 1$.

4. Experiments

Although our algorithm can address a variety of applications, in this paper we will focus on style transfer.

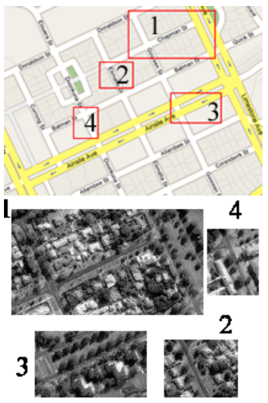
Style is an important factor that our perceptual systems routinely identify from input objects [*e.g.* 10]. For example, we can identify a speaker from his speech, or a dancer from her style of dancing. Here we concentrate on the following style transfer problem: given an unfiltered source image A and some parts of A which are, say, painted by an artist, can we transfer the style of painting onto a new target image B? This problem is applicable to image manipulation software, where the system can learn a style filter for a particular task from a few examples.

Arguably, the super resolution experiment we presented as examples in Figure 1 and 3 is also an instance of style transfer. Here, we are given an image of a forest and some parts of the image transformed via the application of a texture map. The task is to apply the texture map consistently to the rest of the image. As shown in Figure 3, our algorithm delivers visually pleasing results which are faithful to the original texture map.

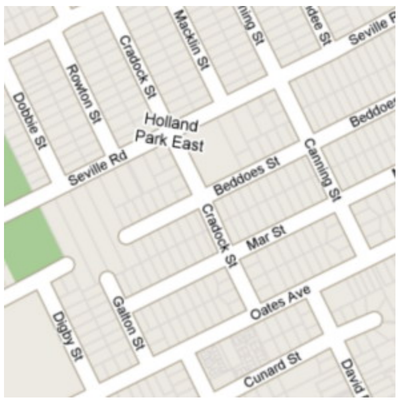
The next experiment shows a scenario where styles are intended to be transferred from *different* sources. In Figure 4, given the input (a) and the partially labeled target (b), where the target style of sky is different from that of the grass and the trees, our algorithm is able to produce a transformed image (c) which mingles these styles in a visually consistent way. Similar results are obtained when we modify the target styles: B' in (e) is produced when using styles in (d) where, again, the target styles of sky, grass, and trees come from two different sources. Image analogies still produces visually inferior result as evidenced in (f).

Our final style transfer experiment takes a girl image as input A (Figure 6a) where only a small fraction has access to the target fabric style as A' (Figure 6b). When apply to the same girl image B, comparing to other methods, clearly the output of our method (panel (c)) is visually more pleasing and consistent with the transferred style. Two more examples are presented in Figure 7 to illustrate the scenario where entirely different images are used as B, the fabric style can still be consistently transferred.

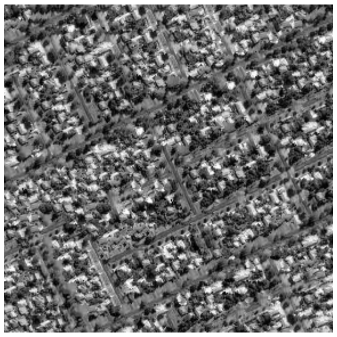
4.1. Fantasizing Satellite Images



(a) A and A' of Braddon



(b) B: Brisbane map


(c) B' of our algorithm on Brisbane

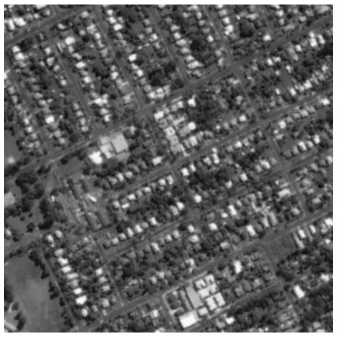

(d) Ground truth of B'

Figure 5: An example of fantasizing satellite maps.

We now report results on a novel experiment. Given a map and its corresponding satellite image, we use our algorithm to fantasize the satellite image given a map. This is useful in applications like google maps, where we might have some archived maps but the satellite images might not be entirely available. Fantasizing these images can arguably provide perceptually consistent prediction of the missing satellite images. A similar application is called texture-by-number in [5].

To test the efficacy of our algorithm on this problem we trained it on maps and the corresponding satellite images of Braddon, a suburb in ACT, Australia, and applied it to fantasize satellite images of a suburb in Brisbane, Australia. Results are shown in Figure 5. Surprisingly, the fantasized map is visually very consistent and plausible, even though it is not close to the ground truth as presented in (d).

5. Outlook and Discussion

We presented an algorithm which produces visually consistent image analogies by employing modern tools from machine learning. In particular, we use semi-supervised learning to exploit unlabeled samples, and a graphical model to ensure global consistency. Results on a wide range of images demonstrate the efficacy of our method. Future research directions include applications in video, and incorporating active learning in the selection of target parts.

We do note that each stage of our algorithm has tuning parameters. Fortunately, the output of our algorithm is fairly independent of the choice of these parameters, and only moderate tuning effort is needed to produce visually pleasing output.

While graphical models, especially MRFs have a rich history of application in computer vision, it is only recently that applications of semi-supervised learning have emerged. We hope that our research will contribute to popularizing semi-supervised learning in the computer vision community.

Acknowledgements

We thank Dale Schuurmans for constructive discussions. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program. This work is supported by the IST Program of the European Community, under the Pascal Network of Excellence, IST-2002-506778.

References

[1] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.*, 7:2399–2434, November 2006. 3

[2] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proc. SIGGRAPH*, pages 341–346, 2001. 2, 5

[3] W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. *IEEE Computer Graphics and Applications*, 22(2):56–65, 2002. 2, 4

[4] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *Int. J. Comput. Vision*, 40(1):25–47, 2000. 2, 4, 5

[5] A. Hertzmann, C. Jacobs, N. Oliver, B. Curless, and D. Salesin. Image analogies. In *Proc. SIGGRAPH*, pages 327–340, 2001. 1, 2, 4, 6

[6] M. I. Jordan. *An Introduction to Probabilistic Graphical Models*. MIT Press, 2002. To Appear. 4

[7] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *Proc. SIGGRAPH*, pages 689–694, 2004. 5

[8] R. Rosales, K. Achan, and B. Frey. Unsupervised image translation. *iccv*, 01:472, 2003. 2

[9] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *Proc. Intl. Conf. Machine Learning*, pages 824–831, 2005. 4, 7

[10] J. Tenenbaum and W. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000. 6

[11] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, University of Wisconsin-Madison, 2005. 3

6. Appendix

6.1. Proof of Equation (3)

The proof we present below is similar to the proof in [9]. But to make it self-contained, we use our own notation and it is a little simplified compared with [9]. We also try to avoid the tricky argument needed for positive *semi*-definition inner product in [9].

Restatement of the conclusion:

Given points $x_i \in \mathcal{X}$ for $i \in [n] := 1, ..., n$, and an RKHS $\mathcal{H} := \left(A, \langle\cdot,\cdot\rangle_{\mathcal{H}}\right)$ with corresponding kernel function $k(\cdot,\cdot)$. Let $\mathcal{V}$ be a linear space with positive *semi*-definite inner product and let $S : \mathcal{H} \to \mathcal{V}$ be a bounded linear operator.

Then $\tilde{\mathcal{H}} := \left(A, \langle\cdot,\cdot\rangle_{\tilde{\mathcal{H}}}\right)$ is also an RKHS under $\langle f, g\rangle_{\tilde{\mathcal{H}}} := \langle f, g\rangle_{\mathcal{H}} + \langle Sf, Sg\rangle_{\mathcal{V}}$.

Moreover, if $\langle Sf, Sg\rangle_{\mathcal{V}} = \mathbf{f}^\top M\,\mathbf{g}$, where $\mathbf{f} = \left[f(x_1), ..., f(x_n)\right]^\top$ (similarly $\mathbf{g}$) and M is a constant $n \times n$ positive *semi*-definite matrix (*e.g.* graph Laplacian L), then the kernel function $\tilde{k}$ of $\tilde{\mathcal{H}}$ can be written as $\tilde{k}(x, x') = k(x, x') - \mathbf{k}_x^\top (I + MK)^{-1} M\,\mathbf{k}_{x'}$, where $\mathbf{k}_x = \left[k(x_1, x), ..., k(x_n, x)\right]^\top$.

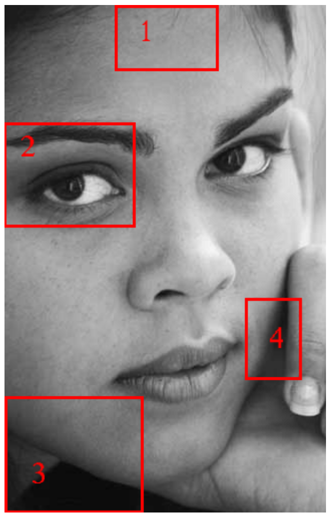
Proof. Since $\|f\|_{\mathcal{H}} \le \|f\|_{\tilde{\mathcal{H}}}$ for all $f \in A$, any Cauchy sequence in the norm $\|\cdot\|_{\tilde{\mathcal{H}}}$ must also be Cauchy in the norm $\|\cdot\|_{\mathcal{H}}$. So completeness of $\mathcal{H}$ implies completeness of $\tilde{\mathcal{H}}$, and $\tilde{\mathcal{H}}$ is Hilbert. Furthermore, since $\mathcal{H}$ is RKHS, there is a $C \in \mathbb{R}^+$ such that for any $x \in \mathcal{X}$ and $f \in A$, $|f(x)| \le C\|f\|_{\mathcal{H}} \le C\|f\|_{\tilde{\mathcal{H}}}$. Hence $\tilde{\mathcal{H}}$ is also RKHS. Denote its corresponding kernel function as $\tilde{k}(\cdot,\cdot)$.

Let $\mathcal{H}^{\|} := span\left\{k(x_i,\cdot)| \in [n]\right\}$, and $\tilde{\mathcal{H}}^{\|} := span\left\{\tilde{k}(x_i,\cdot)\middle| i \in [n]\right\}$. Suppose their orthogonal spaces are $\mathcal{H}^\perp$ and $\tilde{\mathcal{H}}^\perp$ respectively. Since $\left\langle f, \tilde{k}(x_i,\cdot)\right\rangle_{\tilde{\mathcal{H}}} = f(x_i) = \langle f, k(x_i,\cdot)\rangle_{\mathcal{H}}$ for all $f \in A$, so $\left\langle f, \tilde{k}(x_i,\cdot)\right\rangle_{\tilde{\mathcal{H}}} = 0 \Leftrightarrow \langle f, k(x_i,\cdot)\rangle_{\mathcal{H}} = 0$. Hence $\tilde{\mathcal{H}}^\perp = \mathcal{H}^\perp$, and thus $\mathcal{H}^{\|} = \tilde{\mathcal{H}}^{\|}$.

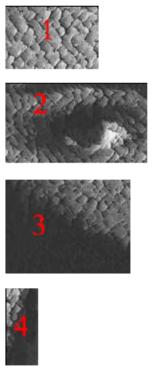
For any $f \in \mathcal{H}^\perp = \tilde{\mathcal{H}}^\perp$, we have $f(x_i) = 0$ for all $i \in [n]$. Thus $\|Sf\|_{\mathcal{V}}^2 = \mathbf{f}^\top M\,\mathbf{f} = 0$, and so $\langle Sf, Sg\rangle_{\mathcal{V}} = \mathbf{f}^\top M\,\mathbf{g} = 0$ for any $g \in A$. Hence for any $x \in \mathcal{X}$,

$$\langle f, k(x,\cdot)\rangle_{\mathcal{H}} = f(x) = \left\langle f, \tilde{k}(x,\cdot)\right\rangle_{\tilde{\mathcal{H}}} \\ = \left\langle f, \tilde{k}(x,\cdot)\right\rangle_{\mathcal{H}} + \left\langle Sf, S\tilde{k}(x,\cdot)\right\rangle_{\mathcal{V}} = \left\langle f, \tilde{k}(x,\cdot)\right\rangle_{\mathcal{H}}.$$

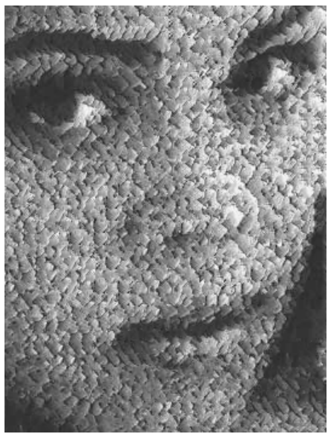
So $\tilde{k}(x,\cdot) - k(x,\cdot) \in \mathcal{H}^{\|}$, *i.e.* there are $\beta_i(x) \in \mathbb{R}$, s.t. $\tilde{k}(x,\cdot) = k(x,\cdot) + \sum_i \beta_i(x)\,k(x_i,\cdot)$. (*)



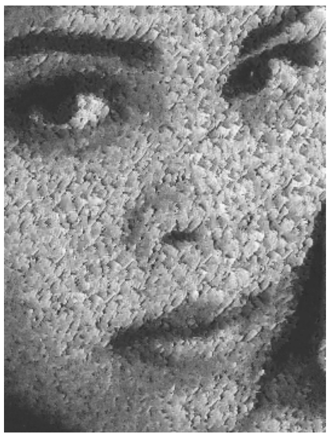
(a) A and B



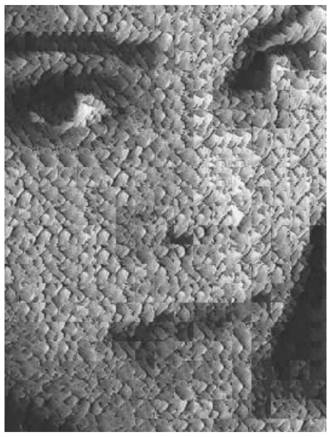
(b) A'



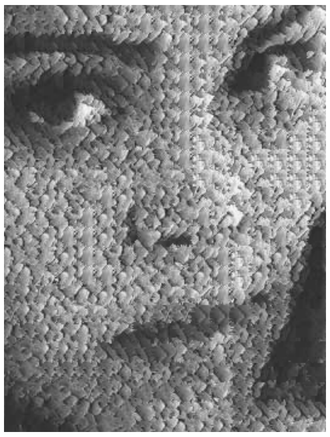
(c) B' by our algorithm



(d) B' by image analogies

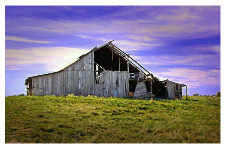


(e) B' by only inference

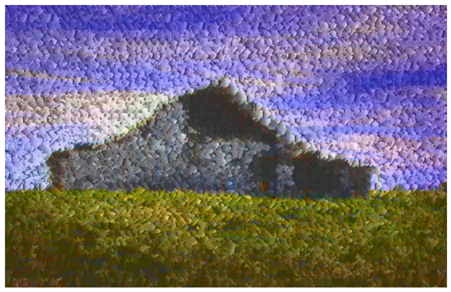


(f) B' by only inference and quilting

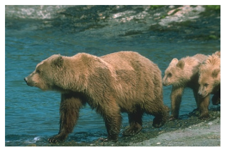
Figure 6: A style transfer example (to be continued).



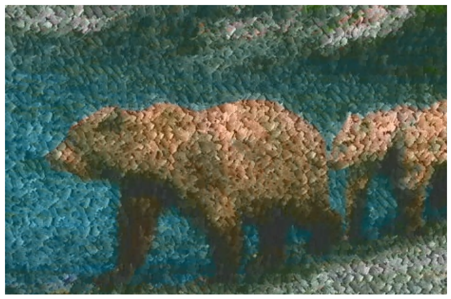
(a) B



(b) B' by our algorithm



(c) B



(d) B' by our algorithm

Figure 7: (Cont. from Figure 6) Take images (a) and (c) as B, respectively.

To calculate $\beta_i(x)$, just observe

$$\begin{aligned} k(x_i, x) &= \left\langle k(x_i, \cdot), \tilde{k}(x, \cdot) \right\rangle_{\tilde{\mathcal{H}}} \\ &= \left\langle k(x_i, \cdot), k(x, \cdot) + \sum_i \beta_i(x) k(x_i, \cdot) \right\rangle_{\tilde{\mathcal{H}}} \\ &= \left\langle k(x_i, \cdot), k(x, \cdot) + \sum_j \beta_j(x) k(x_j, \cdot) \right\rangle_{\mathcal{H}} + \mathbf{k}_{x_i}^\top M\gamma, \end{aligned}$$

where the j^{th} element of vector γ is $\gamma_j = k(x, x_j) + \sum_i \beta_i(x) k(x_i, x_j)$. Enumerating $x = x_i$ over $i \in [n]$ and denoting $\beta(x) = [\beta_1(x), ..., \beta_n(x)]^\top$, we have $(I + MK)\beta(x) = -M\,\mathbf{k}_x$, where K is the Gram matrix ($K_{ij} = k(x_i, x_j)$). Substituting $\beta(x) = -(I + MK)^{-1} M\,\mathbf{k}_x$ back into (*), we obtain $\tilde{k}(x, x') = k(x, x') - \mathbf{k}_x^\top (I + MK)^{-1} M\,\mathbf{k}_{x'}$. □