Journal of Machine Learning Research 8 (2007) 725-760 Submitted 9/05; Revised 1/07; Published 4/07

The Pyramid Match Kernel: Efficient Learning with Sets of Features

Kristen Grauman GRAUMAN@CS.UTEXAS.EDU
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712, USA

Trevor Darrell TREVOR@CSAIL.MIT.EDU
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Editor: Pietro Perona

Abstract

In numerous domains it is useful to represent a single example by the set of the local features or parts that comprise it. However, this representation poses a challenge to many conventional machine learning techniques, since sets may vary in cardinality and elements lack a meaningful ordering. Kernel methods can learn complex functions, but a kernel over unordered set inputs must somehow solve for correspondences—generally a computationally expensive task that becomes impractical for large set sizes. We present a new fast kernel function called the *pyramid match* that measures partial match similarity in time linear in the number of features. The pyramid match maps unordered feature sets to multi-resolution histograms and computes a weighted histogram intersection in order to find implicit correspondences based on the finest resolution histogram cell where a matched pair first appears. We show the pyramid match yields a Mercer kernel, and we prove bounds on its error relative to the optimal partial matching cost. We demonstrate our algorithm on both classification and regression tasks, including object recognition, 3-D human pose inference, and time of publication estimation for documents, and we show that the proposed method is accurate and significantly more efficient than current approaches.

Keywords: kernel, sets of features, histogram intersection, multi-resolution histogram pyramid, approximate matching, object recognition

1. Introduction

In a variety of domains, it is often natural and meaningful to represent a data object with a collection of its parts or component features. For instance, in computer vision, an image may be described by local features extracted from patches around salient interest points, or a shape may be described by local descriptors defined at edge pixels. Likewise, in natural language processing, documents or topics may be represented by sets or bags of words; in computational biology, a disease may be characterized by sets of gene-expression data from multiple patients. In such cases, one set of feature vectors denotes a single instance of a particular class of interest (an object, shape, document, etc.). The number of features per example varies, and within a single instance the component features may have no inherent ordering.

Classification and regression with these sets (or bags) of features is challenging. Kernel-based learning methods are appealing for their generalization ability and efficiency, however conventional

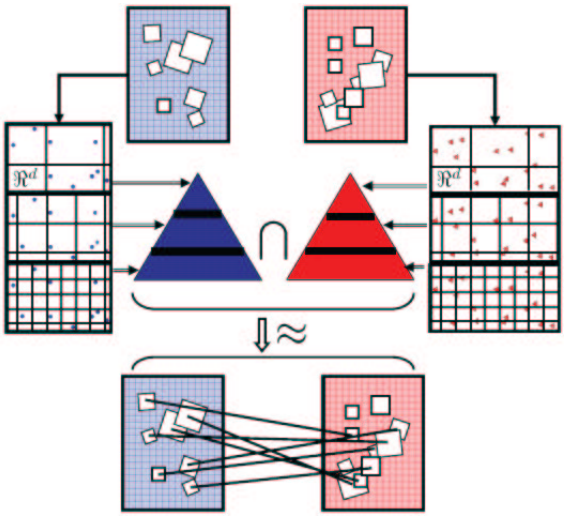


Figure 1: The pyramid match intersects histogram pyramids formed over sets of features, approximating the optimal correspondences between the sets' features. For example, vectors describing the appearance or shape within local image patches can be used to form a feature set for each image; the pyramid match approximates the similarity according to a partial matching in that feature space. (The feature space can be any local description of a data object, images or otherwise; for images the features typically will *not* be the spatial image coordinates.)

kernels are designed to operate on fixed-length vector inputs, where each dimension corresponds to a particular global attribute for that instance; the commonly used general-purpose kernels defined on $\Re^n$ inputs are not applicable in the space of vector sets. Existing kernel-based approaches specially designed for matching sets of features generally require either solving for explicit correspondences between features (which is computationally costly and prohibits the use of large inputs) or fitting a particular parametric distribution to each set (which makes restrictive assumptions about the data and can also be computationally expensive).

In this work we present the *pyramid match kernel*—a new kernel function over unordered feature sets that allows them to be used effectively and efficiently in kernel-based learning methods. Each feature set is mapped to a multi-resolution histogram that preserves the individual features' distinctness at the finest level. The histogram pyramids are then compared using a weighted histogram intersection computation, which we show defines an implicit correspondence based on the finest resolution histogram cell where a matched pair first appears (see Figure 1).

The similarity measured by the pyramid match approximates the similarity measured by the optimal correspondences between feature sets of unequal cardinality (i.e., the *partial matching* that optimally maps points in the lower cardinality set to some subset of the points in the larger set, such that the sum of the distances between matched points is minimized). Our kernel is extremely efficient and can be computed in time that is linear in the sets' cardinality. We show that the kernel function is positive-definite, meaning that it is appropriate to use with learning methods that guarantee convergence to a unique optimum only for positive-definite kernels (e.g., the support vector

machine). We also provide theoretical approximation bounds for the pyramid match cost relative to the optimal partial matching cost.

Because it does not penalize the presence of superfluous data points, the proposed kernel is robust to clutter. As we will show, this translates into the ability to handle common issues faced in vision tasks like object recognition or pose estimation: unsegmented images, poor segmentations, varying backgrounds, and occlusions. The kernel also respects the co-occurrence relations inherent in the input sets: rather than matching features in a set individually, ignoring potential dependencies conveyed by features within one set, our similarity measure captures the features' joint statistics.

Other approaches to this problem have recently been proposed (Wallraven et al. 2003; Lyu 2005; Boughhorbel et al. 2004; Kondor and Jebara 2003; Wolf and Shashua 2003; Moreno et al. 2003; Shashua and Hazan 2005; Cuturi and Vert 2005; Boughorbel et al. 2005; Lafferty and Lebanon 2002), but unfortunately each suffers from some number of the following drawbacks: computational complexities that make large feature set sizes infeasible; limitations to parametric distributions which may not adequately describe the data; kernels that are not positive-definite; limitations to sets of equal size; and failure to account for dependencies within feature sets.

Our method addresses each of these issues, resulting in a kernel appropriate for comparing unordered, variable-sized feature sets within any existing kernel-based learning paradigm. We demonstrate our algorithm in a variety of classification and regression tasks: object recognition from sets of image patch features, 3-D human pose inference from sets of local contour features from monocular silhouettes, and documents' time of publication estimation from bags of local latent semantic features. The results show that the proposed approach achieves an accuracy that is comparable to or better than that of state-of-the-art techniques, while requiring significantly less computation time.

2. Related Work

In this section, we review relevant work on learning with sets of features, using kernels and support vector machines (SVMs) for recognition, and multi-resolution image representations.

Kernel-based learning algorithms, which include SVMs, kernel PCA, and Gaussian Processes, have become well-established tools that are useful in a variety of contexts, including discriminative classification, regression, density estimation, and clustering (Shawe-Taylor and Cristianini, 2004; Vapnik, 1998; Rasmussen and Williams, 2006). However, conventional kernels (such as the Gaussian RBF or polynomial) are designed to operate on $\Re^n$ vector inputs, where each vector entry corresponds to a particular global attribute for that instance. As a result, initial approaches using SVMs for recognition were forced to rely on global image features—ordered features of equal length measured from the image as a whole, such as color or grayscale histograms or vectors of raw pixel data (Chapelle et al., 1999; Roobaert and Hulle, 1999; Odone et al., 2005). Such global representations are known to be sensitive to real-world imaging conditions, such as occlusions, pose changes, or image noise.

Recent work has shown that local features invariant to common image transformations (e.g., SIFT, Lowe, 2004) are a powerful representation for recognition, because the features can be reliably detected and matched across instances of the same object or scene under different viewpoints, poses, or lighting conditions. Most approaches, however, perform recognition with local feature representations using nearest-neighbor (e.g., Belongie et al., 2002; Grauman and Darrell, 2004; Sivic and Zisserman, 2003; Berg et al., 2005) or voting-based classifiers followed by an alignment

step (e.g., Lowe, 2004; Mikolajczyk and Schmid, 2001); both may be impractical for large training sets, since their classification times increase with the number of training examples. A support vector classifier or regressor, on the other hand, identifies a sparse subset of the training examples (the support vectors) to delineate a decision boundary or approximate a function of interest.

In order to more fully leverage existing kernel-based learning tools for situations where the data cannot be naturally represented by a Euclidean vector space—such as graphs, strings, or trees—researchers have developed specialized similarity measures (Gartner, 2003). In fact, due to the increasing prevalence of data that is best represented by sets of local features, several researchers have recently designed kernel functions that can handle unordered sets as input (Lyu 2005; Kondor and Jebara 2003; Wolf and Shashua 2003; Shashua and Hazan 2005; Boughhorbel et al. 2004; Boughorbel et al. 2005; Wallraven et al. 2003; Cuturi and Vert 2005; Moreno et al. 2003; Lafferty and Lebanon 2002). Nonetheless, current approaches are either prohibitively computationally expensive, are forced to make assumptions regarding the parametric form of the features, discard information by replacing inputs with prototypical features, ignore important co-occurrence information by considering features independently, are not positive-definite, and (or) are limited to sets of equal size. In addition, none have shown the ability to learn a real-valued function from sets of features; results have only been shown for classification tasks. See Figure 2 for a concise comparison of the approaches.

Approaches which fit a parametric model to feature sets in order to compare their distributions (Kondor and Jebara, 2003; Moreno et al., 2003; Cuturi and Vert, 2005; Lafferty and Lebanon, 2002) can be computationally costly and have limited applicability, since they assume both that features within a set will conform to the chosen distribution, and that sets will be adequately large enough to extract an accurate estimate of the distribution's parameters. These assumptions are violated regularly by real data, which will often exhibit complex variations within a single bag of features (e.g., patches from an image), and will produce wide ranges of cardinalities per instance (e.g., titles of documents have just a few word features). Our method instead takes a non-parametric, "model-free" approach, representing sets of features directly with multi-dimensional, multi-resolution histograms.

Kernel methods that use explicit correspondences between two sets' features search one set for the best matching feature for each member in the other, and then define set similarity as a function over those component similarity values (Wallraven et al., 2003; Lyu, 2005; Boughhorbel et al., 2004; Boughorbel et al., 2005). These methods have complexities that are quadratic in the number of features, hindering usage for kernel-based learning when feature sets are large. The "intermediate" matching kernel of Boughorbel et al. (2005) has a quadratic run-time if the number of prototypes $p = O(m)$. That is reduced if p is set so that $p < m$; however, the authors note that higher values of p yield more accurate results. Furthermore, matching each input feature independently ignores useful information about intra-set dependencies. In contrast, our kernel captures the joint statistics of co-occurring features by matching them concurrently as a set.

In the method of Wolf and Shashua (2003), similarity is measured in terms of the principal angle between the linear subspaces spanned by two sets' vector elements; the kernel has a cubic complexity and is only positive-definite for sets of equal cardinality. In the work of Shashua and Hazan (2005), an algebraic kernel is used to combine similarities given by local (vector-based) kernels, with the weighting chosen to reflect whether the features are in alignment (ordered). When set cardinalities vary, inputs are padded with zeros so as to form equal-size matrices; results are only shown for a classification task with input sets whose features' ordering is known.

Method	Complexity	Captures co-occurrences	Positive-definite	Non-parametric	Handles unequal cardinalities
Match	$O(dm^2)$			x	x
Exponent match	$O(dm^2)$		x	x	x
Greedy match	$O(dm^2)$	x		x	x
Principal angles	$O(dm^3)$	x	x		
Intermediate	$O(dpm)$		x	x	x
Bhattacharyya's	$O(dm^3)$	x	x		x
KL-divergence	$O(dm^2)$	x			x
Pyramid match	$O(dm \log D)$	x	x	x	x

Figure 2: Comparing the properties of kernel approaches to matching unordered sets. "Match" refers to the kernel of Wallraven et al. (2003), "Exponent match" is the kernel of Lyu (2005), "Greedy match" is from Boughhorbel et al. (2004), "Principal angles" is from Wolf and Shashua (2003), "Intermediate" is from Boughorbel et al. (2005), "Bhattacharyya's" is from Kondor and Jebara (2003), and "KL-divergence" is from Moreno et al. (2003). "Pyramid match" refers to the proposed kernel. Each method's computational cost is for computing a single kernel value. d is vector dimension, m is maximum set cardinality, p is the number of prototype features used by Boughorbel et al. (2005), and D is the value of the maximal feature range.

Several computer vision researchers have transformed the set of real-valued feature vectors coming from one image into a single flat histogram that counts the frequency of occurrence of some number of pre-defined (quantized) feature prototypes. In this way the quantized feature space provides a *visual vocabulary* or *bag-of-words* vector representation which can be used in conjunction with some vector-based kernels or similarity measures. This type of representation was explored for texture recognition using nearest-neighbors (Leung and Malik, 2001; Hayman et al., 2004), and more recently has been shown for classification of object categories using SVMs, Naïve Bayes classifiers, and a probabilistic Latent Semantic Analysis framework (Csurka et al., 2004; Willamowski et al., 2004; Sivic et al., 2005).

The bag-of-words is appealing because it allows existing vector-based methods to be applied and can describe the overall distribution of features in an image. However, this representation faces the substantial challenge of generating an appropriately descriptive quantization of the feature space. Bin boundary issues have been shown to create matching problems for flat histograms (Rubner et al., 2000), and though the right size of a vocabulary for a given data set can be critical to recognition performance (Csurka et al., 2004; Grauman and Darrell, 2004) it must still be determined empirically. Generating the vocabulary from large amounts of data is generally computationally costly, and it is not clear whether a generic or universal feature quantization is more or less effective than data set-dependent vocabularies. Finally, unlike the approach we develop here, existing bag-of-words techniques can only compare entire images to one another, do not allow partial matchings, and cannot be used to extract correspondences telling which features match to which. In our experiments we find that the pyramid match outperforms the bag-of-words approach for object category recognition on a challenging data set (see Section 9).

An alternative approach to discriminative classification when dealing with unordered set data is to designate prototypical examples from each class, and then represent examples by a vector giving their distances to each prototype; standard algorithms that handle vectors in a Euclidean space are then applicable. Zhang and Malik (2003) build such a classifier for handwritten digits, and use the shape context distance of Belongie et al. (2002) as the measure of similarity. Shan et al. also explore a representation based on distances to prototypes in order to avoid feature matching when recognizing vehicles (Shan et al., 2005). The issues faced by such a prototype-based method are determining which examples should serve as prototypes, choosing how many there should be, and updating the prototypes properly when new types of data are encountered. The method of Holub et al. (2005a) uses a hybrid generative-discriminative approach for object recognition, combining the Fisher kernel (Jaakkola and Haussler, 1999) and a probabilistic constellation model.

Our feature representation is based on a multi-resolution histogram, or pyramid, which is computed by binning data points into discrete regions of increasingly larger size. Single-level histograms have been used in various visual recognition systems, one of the first being that of Swain and Ballard (1991), where the intersection of global color histograms was used to compare images. Pyramids have been shown to be a useful representation in a wide variety of image processing tasks, from image coding (Burt and Adelson, 1983), to optical flow (Anandan, 1987), to texture modeling (Malik and Perona, 1990). See work by Hadjidemetriou et al. (2004) for a summary.

In the method of Indyk and Thaper (2003), multi-resolution histograms are compared with L_1 distance to approximate a least-cost matching of equal-mass global color histograms for nearest neighbor image retrievals. This work inspired our use of a similar representation for point sets and to consider counting matches within histograms. However, in contrast Indyk and Thaper's approach, our method builds a discriminative classifier or regressor over sets of local features, and it allows inputs to have unequal cardinalities. Most importantly, it enables partial matchings, which is important in practice for handling clutter and unsegmented images. In addition, we show that our approximate matching forms a valid Mercer kernel and explore its use for kernel-based learning for various applications.

In this work we develop a new kernel function that efficiently handles inputs that are unordered sets of varying sizes. We show how the pyramid match kernel may be used in conjunction with existing kernel-based learning algorithms to successfully learn decision boundaries or real-valued functions from the multi-set representation. Ours is the first work to show the histogram pyramid's connection to the optimal partial matching when used with a hierarchical weighted histogram intersection similarity measure. [1]

3. Approach

The main contribution of this work is a new kernel function based on implicit correspondences that enables discriminative classification and regression for unordered, variable-sized sets of vectors. The kernel is provably positive-definite. The main advantages of our algorithm are its efficiency, its use of the joint statistics of co-occurring features, and its resistance to clutter or "superfluous"

1. We first introduced the pyramid match kernel for the purpose of discriminative object recognition in a recent conference paper (Grauman and Darrell, 2005). This paper builds on that initial work: here we also demonstrate its utility for regression, give results on additional new data sets (one of which is non-vision), provide a more in-depth description of the kernel, and prove bounds on the error of the pyramid match relative to the optimal partial matching cost.

data points. The basic idea of our method is to map sets of features to multi-resolution histograms, and then compare the histograms with a weighted histogram intersection measure in order to approximate the similarity of the best partial matching between the feature sets. We call the proposed matching kernel the *pyramid match kernel* because input sets are converted to multi-resolution histograms.

3.1 Preliminaries

We consider a feature space F of d-dimensional vectors. The point sets (or multi-sets, since duplications of features may occur within a single set) we match will come from the input space S, which contains sets of feature vectors drawn from F:

$$S = \Big\{ \mathbf{X} | \mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \Big\},$$

where each feature is a d-dimensional vector, $\mathbf{x}_i \in F \subseteq \Re^d$, and $m = |\mathbf{X}|$. Note that the point dimension d is fixed for all features in F, but the value of m may vary across instances in S. The values of elements in vectors in F have a maximal range D, and the minimum inter-vector distance between unique points is 1, which may be enforced by scaling the data to some precision and truncating to integer values.

In this work we want to efficiently approximate the optimal partial matching. A partial matching between two point sets is an assignment that maps all points in the smaller set to some subset of the points in the larger (or equally-sized) set. Given point sets $\mathbf{X}$ and $\mathbf{Y}$, where $m = |\mathbf{X}|$, $n = |\mathbf{Y}|$, and $m \leq n$, a partial matching

$$\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi) = \{(\mathbf{x}_1, \mathbf{y}_{\pi_1}), \ldots, (\mathbf{x}_m, \mathbf{y}_{\pi_m})\}$$

pairs each point in $\mathbf{X}$ to some unique point in $\mathbf{Y}$ according to the permutation of indices specified by $\pi = [\pi_1, \ldots, \pi_m]$, $1 \leq \pi_i \leq n$, where π_i specifies which point $\mathbf{y}_{\pi_i} \in \mathbf{Y}$ is matched to $\mathbf{x}_i \in \mathbf{X}$, for $1 \leq i \leq m$. The cost of a partial matching is the sum of the distances between matched points:

$$\mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi)) = \sum_{\mathbf{x}_i \in \mathbf{X}} ||\mathbf{x}_i - \mathbf{y}_{\pi_i}||_1.$$

The optimal partial matching $\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)$ uses the assignment π^* that minimizes this cost:

$$\pi^* = \underset{\pi}{\operatorname{argmin}}\, \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi)). \tag{1}$$

In order to form a kernel function based on correspondences, we are interested in evaluating partial matching *similarity*, where similarity is measured in terms of inverse distance or cost. The similarity $\mathcal{S}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi))$ of a partial matching is the sum of the inverse distances between matched points:

$$\mathcal{S}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi)) = \sum_{\mathbf{x}_i \in \mathbf{X}} \frac{1}{||\mathbf{x}_i - \mathbf{y}_{\pi_i}||_1 + 1},$$

where the distance in the denominator is incremented by 1 to avoid division by zero. We define the optimal partial matching similarity score to be the similarity resulting from the same matching that minimizes the cost (Eqn. 1).

3.2 The Pyramid Match Algorithm

The pyramid match approximation uses a multi-dimensional, multi-resolution histogram pyramid to partition the feature space into increasingly larger regions. At the finest resolution level in the pyramid, the partitions (bins) are very small; at successive levels they continue to grow in size until the point where a single partition encompasses the entire feature space. At some level along this gradation in bin sizes, any two points from any two point sets will begin to share a bin, and when they do, they are considered matched. The pyramid allows us to extract a matching score without computing distances between any of the points in the input sets—when points sharing a bin are counted as matched, the size of that bin indicates the farthest distance any two points in it could be from one another.

Each feature set is mapped to a multi-resolution histogram that preserves the individual features' distinctness at the finest level. The histogram pyramids are then compared using a weighted histogram intersection computation, which we show defines an implicit partial correspondence based on the finest resolution histogram cell where a matched pair first appears. The computation time of both the pyramids themselves as well as the weighted intersection is linear in the number of features.

The feature extraction function Ψ for an input set $\mathbf{X}$ is defined as:

$$\Psi(\mathbf{X}) = [H_0(\mathbf{X}), \ldots, H_{L-1}(\mathbf{X})], \tag{2}$$

where $\mathbf{X} \in S$, $L = \lceil \log_2 D \rceil + 1$, $H_i(\mathbf{X})$ is a histogram vector formed over points in $\mathbf{X}$ using d-dimensional bins of side length 2^i, and $H_i(\mathbf{X})$ has a dimension $r_i = \left(\frac{D}{2^i}\right)^d$. In other words, $\Psi(\mathbf{X})$ is a histogram pyramid, where each subsequent component histogram has bins that double in size (in all d dimensions) compared to the previous one. The bins in the finest-level histogram H_0 are small enough that each unique d-dimensional data point from features in F falls into its own bin, and then the bin size increases until all points in F fall into a single bin at level $L-1$.[2]

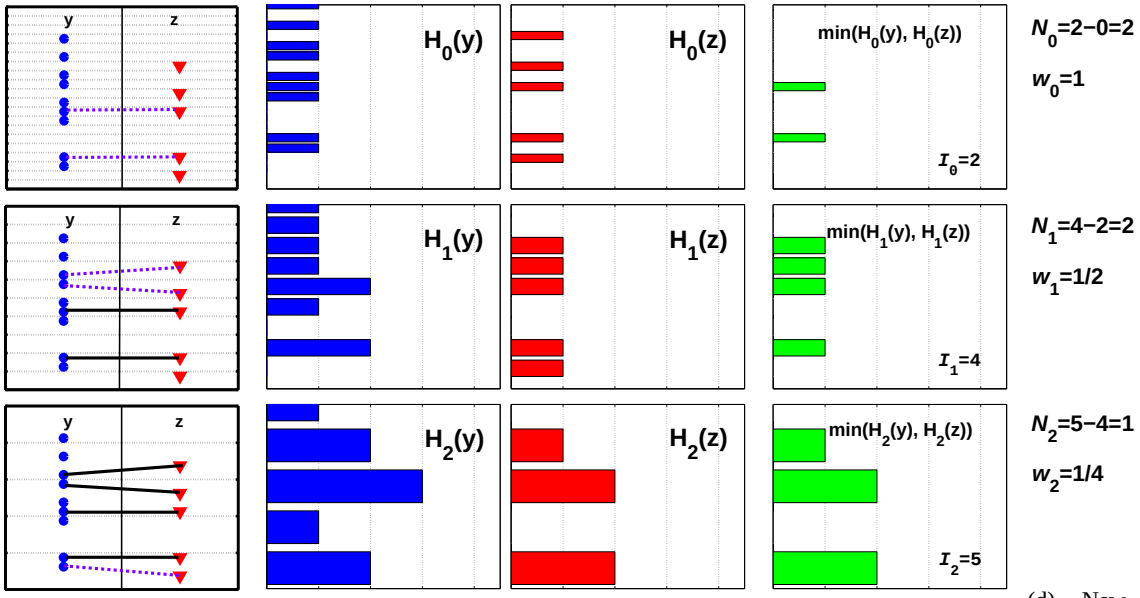
The pyramid match $\mathcal{P}_\Delta$ measures similarity (or dissimilarity) between point sets based on implicit correspondences found within this multi-resolution histogram space. The similarity between two input sets $\mathbf{Y}$ and $\mathbf{Z}$ is defined as the weighted sum of the number of feature matchings found at each level of the pyramid formed by Ψ:

$$\mathcal{P}_\Delta(\Psi(\mathbf{Y}), \Psi(\mathbf{Z})) = \sum_{i=0}^{L-1} w_i N_i, \tag{3}$$

where N_i signifies the number of newly matched pairs at level i, and w_i is a weight for matches formed at level i (and will be defined below). A new match is defined as a pair of features that were not in correspondence at any finer resolution level.

The matching approximation implicitly finds correspondences between point sets, if we consider two points matched once they fall into the same histogram bin, starting at the finest resolution level where each unique point is guaranteed to be in its own bin. The correspondences are *implicit* in that matches are counted and weighted according to their strength, but the specific pairwise links between points need not be individually enumerated. The matching is a hierarchical process: vectors not found to correspond at a fine resolution have the opportunity to be matched at coarser resolutions. For example, in Figure 3, there are two points matched at the finest scale, two new matches at

2. To enable accurate pyramid matching even with high-dimensional feature spaces, we have developed a variant where the pyramid bin structure depends on the distribution of the data (Grauman and Darrell, 2007).



(a) Point sets (b) Histogram pyramids (c) Intersections (d) New matches

Figure 3: The pyramid match ($\mathcal{P}_\Delta$) determines a partial correspondence by matching points once they fall into the same histogram bin. In this example, two 1-D feature sets are used to form two histogram pyramids. Each row corresponds to a pyramid level. In (a), the set $\mathbf{Y}$ is on the left side, and the set $\mathbf{Z}$ is on the right. (Points are distributed along the vertical axis, and these same points are repeated at each level.) Light dotted lines are bin boundaries, bold dashed lines indicate a new pair matched at this level, and bold solid lines indicate a match already formed at a finer resolution level. In (b) multi-resolution histograms are shown, with bin counts along the horizontal axis. In (c) the intersection pyramids between the histograms in (b) are shown. $\mathcal{P}_\Delta$ uses these intersection counts to measure how many new matches occurred at each level. Here, $I_i = I(H_i(\mathbf{Y}), H_i(\mathbf{Z})) = 2, 4, 5$ across levels, and therefore the number of new matches found at each level are $N_i = 2, 2, 1$. ($I_{-1} = 0$ by definition.) The sum over N_i, weighted by $w_i = 1, \frac{1}{2}, \frac{1}{4}$, gives the pyramid match similarity.

the medium scale, and one at the coarsest scale. $\mathcal{P}_\Delta$'s output value reflects the overall similarity of the matching: each newly matched pair at level i contributes a value w_i that is proportional to how similar two points matching at that level must be, as determined by the bin size.

To calculate N_i, the pyramid match makes use of a histogram intersection function I, which measures the "overlap" between two histograms' bin counts:

$$I(\mathbf{A}, \mathbf{B}) = \sum_{j=1}^{r} \min\left(\mathbf{A}^{(j)}, \mathbf{B}^{(j)}\right),$$

where $\mathbf{A}$ and $\mathbf{B}$ are histograms with r bins, and $\mathbf{A}^{(j)}$ denotes the count of the j^{th} bin of $\mathbf{A}$.

Histogram intersection effectively counts the number of points in two sets that match at a given quantization level, that is, fall into the same bin. To calculate the number of newly matched pairs N_i induced at level i, it is sufficient to compute the difference between successive histogram levels' intersections:

$$N_i = I\left(H_i(\mathbf{Y}), H_i(\mathbf{Z})\right) - I\left(H_{i-1}(\mathbf{Y}), H_{i-1}(\mathbf{Z})\right), \tag{4}$$

where H_i refers to the i^{th} component histogram generated by Ψ in Eqn. 2. Note that the measure is not searching explicitly for similar points—it never computes distances between the vectors in each set. Instead, it simply uses the change in intersection values at each histogram level to count the matches as they occur. In addition, due to the subtraction in Eqn. 4, the output score will reflect an underlying matching that is one-to-one.

The sum in Eqn. 3 starts with index $i = 0$ because by the definition of F, the same points that could match at level 0 can match at (a hypothetical) level -1. Therefore we can start collecting new match counts at level 0, and define level -1 to be a base case with no intersections: $I\left(H_{-1}(\mathbf{Y}), H_{-1}(\mathbf{Z})\right) = 0$. All matches formed at level 0 are new.

The number of new matches found at each level in the pyramid is weighted according to the size of that histogram's bins: to measure similarity, matches made within larger bins are weighted less than those found in smaller bins. Specifically, we make a geometric bound of the distance between any two points sharing a particular bin; in terms of L_1 cost, two points in the same bin can only be as far from one another as the sum of the lengths of the bin's sides. At level i in a pyramid, this length is equal to $d2^i$. Thus, the number of new matches induced at level i is weighted by $w_i = \frac{1}{d2^i}$ to reflect the (worst-case) similarity of points matched at that level. Intuitively, this means that similarity between vectors (features in $\mathbf{Y}$ and $\mathbf{Z}$) at a finer resolution—where features are more distinct—is rewarded more heavily than similarity between vectors at a coarser level.[3] Moreover, as we show in Section 6, this particular setting of the weights enables us to prove theoretical error bounds for the pyramid match cost.

From Eqns. 3 and 4, we define the (un-normalized) pyramid match:

$$\tilde{\mathcal{P}}_\Delta\left(\Psi(\mathbf{Y}), \Psi(\mathbf{Z})\right) = \sum_{i=0}^{L-1} w_i \Big(I\left(H_i(\mathbf{Y}), H_i(\mathbf{Z})\right) - I\left(H_{i-1}(\mathbf{Y}), H_{i-1}(\mathbf{Z})\right)\Big), \tag{5}$$

where $\mathbf{Y}, \mathbf{Z} \in S$, and $H_i(\mathbf{Y})$ and $H_i(\mathbf{Z})$ refer to the i^{th} histogram in $\Psi(\mathbf{Y})$ and $\Psi(\mathbf{Z})$, respectively. We normalize this value by the product of each input's self-similarity to avoid favoring larger input sets, arriving at the final kernel value $\mathcal{P}_\Delta\left(\mathbf{P}, \mathbf{Q}\right) = \frac{1}{\sqrt{C}} \tilde{\mathcal{P}}_\Delta\left(\mathbf{P}, \mathbf{Q}\right)$, where $C = \tilde{\mathcal{P}}_\Delta\left(\mathbf{P}, \mathbf{P}\right) \tilde{\mathcal{P}}_\Delta\left(\mathbf{Q}, \mathbf{Q}\right)$.

In order to alleviate quantization effects that may arise due to the discrete histogram bins, we combine the kernel values resulting from multiple (T) pyramid matches formed under different multi-resolution histograms with randomly shifted bins. Each dimension of each of the T pyramids is shifted by an amount chosen uniformly at random from $[0, D]$. This yields T feature mappings $\Psi_1, \ldots, \Psi_T$ that are applied as in Eqn. 2 to map an input set $\mathbf{X}$ to T multi-resolution histograms: $[\Psi_1(\mathbf{X}), \ldots, \Psi_T(\mathbf{X})]$. For inputs $\mathbf{Y}$ and $\mathbf{Z}$, the combined kernel value is then $\sum_{j=1}^{T} \mathcal{P}_\Delta\left(\Psi_j(\mathbf{Y}), \Psi_j(\mathbf{Z})\right)$.

To further refine a pyramid match, multiple pyramids with unique initial (finest level) bin sizes may be used. The set of unique bin sizes produced throughout a pyramid determines the range

3. This is if the pyramid match is a kernel measuring similarity; an inverse weighting scheme is applied if the pyramid match is used to measure cost. To use the matching as a cost function, weights are set as the distance estimates ($w_i = d2^i$); to use as a similarity measure, weights are set as (some function of) the inverse of the distance estimates ($w_i \propto \frac{1}{d2^i}$).

of distances at which features will be considered for possible matchings. With bin sizes doubling in size at every level of a pyramid, this means that the range of distances considered will rapidly increase until the full diameter of the feature space is covered. However, by incorporating multiple pyramids with unique initial bin sizes, we can boost that range of distances to include more diverse gradations. This is accomplished by setting the side length for each histogram $H_i(\mathbf{X})$ to $f2^i$, where f is the side length at the finest resolution (in the above definition, $f = 1$). For example, say $D = 32$, so $L = 6$. Then the set of distances considered with $f = 1$ are $\{1, 2, 4, 8, 16, 32\}$; adding a second pyramid with $f = 3$ increases this set to also consider the distances $\{3, 6, 12, 24\}$. The outputs from pyramids with multiple starting side lengths are combined in the same way that the outputs from pyramids with multiple translations are combined.

In fact, the rationale for considering multiple random shifts of the pyramid grids is not only to satisfy the intuitive need to reduce quantization effects caused by bin placements; it also allows us to theoretically measure how probable it is (in the expectation) that any two points will be separated by a bin boundary, as we describe below in Section 6.

3.3 Partial Match Correspondences

The pyramid match allows input sets to have unequal cardinalities, and therefore it enables partial matchings, where the points of the smaller set are mapped to some subset of the points in the larger set. Dissimilarity is only judged on the most similar part of the empirical distributions, and superfluous data points from the larger set are entirely ignored; the result is a robust similarity measure that accommodates inputs expected to contain extraneous vector entries. This is a common situation when recognizing objects in images, due for instance to background variations, clutter, or changes in object pose that cause different subsets of features to be visible. As a kernel, the proposed matching measure is equipped to handle unsegmented or poorly segmented examples, as we will demonstrate in Section 9.

Since the pyramid match defines correspondences across entire sets simultaneously, it inherently accounts for the distribution of features occurring in one set. In contrast, previous approaches have used each feature in a set to independently index into the second set; this ignores possibly useful information that is inherent in the co-occurrence of a set of distinctive features, and it fails to distinguish between instances where an object has varying numbers of similar features since multiple features may be matched to a single feature in the other set (Wallraven et al., 2003; Lyu, 2005; Boughorbel et al., 2005; Lowe, 2004; Mikolajczyk and Schmid, 2001; Tuytelaars and Gool, 1999; Shaffalitzky and Zisserman, 2002).

4. Efficiency

A key aspect of this method is that we obtain a measure of matching quality between two point sets without computing pairwise distances between their features—an $O(m^2)$ savings over sub-optimal greedy matchings. Instead, we exploit the fact that the points' placement in the pyramid reflects their distance from one another in the feature space.

The time required to compute the L-level histogram pyramid $\Psi(\mathbf{X})$ for an input set with $m = |\mathbf{X}|$ d-dimensional features is $O(dzL)$, where $z = \max(m, k)$ and k is the maximum histogram index value in a single dimension. For a histogram at level i, $k \leq \frac{D}{2^i}$, so at any level, $k \leq D$. (Typically $m > k$.) The bin coordinates corresponding to nonzero histogram entries for each of the $L = \lceil \log_2 D \rceil + 1$ quantization levels are computed directly during a scan of the m input vectors; these entries are

sorted by the bin indices and the bin counts for all entries with the same index are summed to form one entry. This sorting requires only $O(dm+dk)$ time using the radix-sort algorithm with counting sort, a linear time sorting algorithm that is applicable to the integer bin indices Cormen et al. (1990). The histogram pyramid that results is high-dimensional, but very sparse, with only $O(mL)$ nonzero entries that need to be stored.

The computational complexity of $\mathcal{P}_\Delta$ is $O(dmL)$, since computing the intersection values for histograms that have been sorted by bin index requires time linear in the number of nonzero entries (*not* the number of actual bins). With sorted bin index lists, we obtain the intersection value by running one pointer down each of the two lists; one pointer may not advance until the other is incremented down to an index at least as high as the other, or else the end of its list. Then, whenever the two lists share a nonzero index, the intersection count is incremented by the minimum bin count between the two. If one list has a nonzero count for some bin index but the other has no entry for that index, the minimum count is 0, so no update is needed. In the applications we explore in our experiments, typical values of the variables affecting complexity are as follows: $5 \leq d \leq 12$, $300 \leq m \leq 3000$, and $D \approx 250$, which yields $L = 9$.

Generating multiple pyramid matches with randomly shifted grids scales the complexity by T, the constant number of shifts. (Typically we will use $1 \leq T \leq 3$.) All together, the complexity of computing both the pyramids and kernel or cost values is $O(TdmL)$. In contrast, the optimal matching requires $O(dm^3)$ time, which severely limits the practicality of large input sizes. Refer to Figure 2 for complexity comparisons with existing set kernels.

5. Satisfying Mercer's Condition

Kernel-based learning algorithms are founded on the idea of embedding data into a Euclidean space, and then seeking linear relations among the embedded data (Shawe-Taylor and Cristianini, 2004; Vapnik, 1998). For example, a support vector machine (SVM) finds the optimal separating hyperplane between two classes in an embedded space (also referred to as the feature space). A kernel function $K : X \times X \rightarrow \Re$ serves to map pairs of data objects in an input space X to their inner product in the embedding space E, thereby evaluating the similarities between all data objects and determining their relative positions. Linear relations are sought in the embedded space, but the learned function may still be non-linear in the input space, depending on the choice of a feature mapping function $\Phi : X \rightarrow E$.

Only positive semi-definite kernels guarantee an optimal solution to kernel-based algorithms based on convex optimization, which includes SVMs. According to Mercer's theorem, a kernel K is positive semi-definite if and only if there exists a mapping Φ such that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle, \; \forall \mathbf{x}_i, \mathbf{x}_j \in X,$$

where $\langle \cdot, \cdot \rangle$ denotes a scalar dot product. This insures that the kernel corresponds to an inner product in some feature space, where kernel methods can search for linear relationships (Shawe-Taylor and Cristianini, 2004).

Proposition 1

The pyramid match yields a Mercer kernel.

Proof:

Histogram intersection on single resolution histograms over multi-dimensional data was shown to be a positive-definite function by Odone et al. (2005). That is, the intersection $I(H(\mathbf{Y}),H(\mathbf{Z}))$ is a Mercer kernel. The proof shows that there is an explicit feature mapping after which the intersection is an inner product. Specifically, the mapping $\mathcal{V}$ encodes an r-bin histogram H as a p-dimensional binary vector, $p = m \times r$, where m is the total number of points in the histogram:

$$\mathcal{V}(H) = \left(\underbrace{\overbrace{1,\ldots,1}^{H^{(1)}}, \overbrace{0,\ldots,0}^{m-H^{(1)}}}_{\text{first bin}}, \ldots, \underbrace{\overbrace{1,\ldots,1}^{H^{(r)}}, \overbrace{0,\ldots,0}^{m-H^{(r)}}}_{\text{last bin}} \right).$$

The inner product between the binary strings output from $\mathcal{V}$ is equivalent to the original histograms' intersection value (Odone et al., 2005). If m varies across examples, the above holds by setting $p = M \times r$, where M is the maximum size of any input. Note that this binary encoding only serves to prove positive-definiteness and is never computed explicitly.

Using this proof and the closure properties of valid kernel functions, we can show that the pyramid match is a Mercer kernel. The definition given in Eqn. 5 is algebraically equivalent to

$$\tilde{\mathcal{P}}_\Delta(\Psi(\mathbf{Y}),\Psi(\mathbf{Z})) = w_{L-1}\ I(H_{L-1}(\mathbf{Y}),H_{L-1}(\mathbf{Z})) + \sum_{i=0}^{L-2}(w_i - w_{i+1})\ I(H_i(\mathbf{Y}),H_i(\mathbf{Z})),$$

since $I(H_{-1}(\mathbf{Y}),H_{-1}(\mathbf{Z})) = 0$ by definition. Given that Mercer kernels are closed under both addition and scaling by a positive constant (Shawe-Taylor and Cristianini, 2004), the above form shows that the pyramid match kernel is positive semi-definite for any weighting scheme where $w_i \geq w_{i+1}$. In other words, if we can insure that the weights decrease for coarser pyramid levels, then the pyramid match will sum over positively weighted positive-definite kernels, yielding another positive-definite kernel. Using the weights $w_i = \frac{1}{d2^i}$, we do maintain this property. The sum combining the outputs from multiple pyramid matches under different random bin translations likewise remains Mercer. Therefore, the pyramid match is valid for use as a kernel in any existing learning methods that require Mercer kernels.

6. Approximation Error Bounds

In this section we show theoretical approximation bounds on the cost measured by the pyramid match relative to the cost measured by the optimal partial matching defined in Section 3.1. Recall the cost (as opposed to similarity) is measured by setting $w_i = d2^i$ in Eqn. 5. In earlier work, Indyk and Thaper (2003) provided bounds for a multi-resolution histogram embedding's approximation of the optimal bijective matching between inputs with equal cardinality. Some of the main ideas of the proofs below are similar, but they have been adapted and extended to show the expected error bounds for partial matchings, where input sets can have variable numbers of features, and some features do not affect the matching cost.

Proposition 2

For any two point sets $\mathbf{X}$,$\mathbf{Y}$ where $|\mathbf{X}| \leq |\mathbf{Y}|$, we have

$$\mathcal{C}(\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*)) \leq \mathcal{P}_\Delta(\mathbf{X},\mathbf{Y}).$$

Proof:

Consider a matching induced by pairing points within the same cells of each histogram H_i. The cost induced by matching two points within a bin having d sides that are each of length 2^i is at most $d2^i$ under the L_1 ground distance.

There are no pairings induced by the histograms at level -1, so $I(H_{-1}(\mathbf{X}), H_{-1}(\mathbf{Y})) = 0$. At level 0 there are $I(H_0(\mathbf{X}), H_0(\mathbf{Y}))$ pairs of points that can be matched together within the same bins of H_0, which induces a cost no more than $d\ I(H_0(\mathbf{X}), H_0(\mathbf{Y}))$. Then $I(H_1(\mathbf{X}), H_1(\mathbf{Y})) - I(H_0(\mathbf{X}), H_0(\mathbf{Y}))$ new pairs of points are matched at level 1, which induces a cost no more than $2d\,[I(H_1(\mathbf{X}), H_1(\mathbf{Y})) - I(H_0(\mathbf{X}), H_0(\mathbf{Y}))]$, and so on.

In general, histogram H_i induces cost $d2^i\,[I(H_i(\mathbf{X}), H_i(\mathbf{Y})) - I(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y}))]$. Summing the costs induced by all levels, we have:

$$\begin{aligned} C(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) &\leq \sum_{i=0}^{L-1} d2^i \Big(I(H_i(\mathbf{X}), H_i(\mathbf{Y})) - I(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})) \Big) \\ &\leq \mathcal{P}_\Delta(\mathbf{X}, \mathbf{Y}), \end{aligned}$$

where $w_i = d2^i$ in Eqn. 5.

Proposition 3

There is a constant C such that for any two point sets $\mathbf{X}$ and $\mathbf{Y}$, where $|\mathbf{X}| \leq |\mathbf{Y}|$, if the histogram bin boundaries are shifted randomly in the same way when computing $\Psi(\mathbf{X})$ and $\Psi(\mathbf{Y})$, then the expected value of the pyramid match cost is bounded:

$$E[\mathcal{P}_\Delta(\mathbf{X}, \mathbf{Y})] \leq \Big(C \cdot d \log D + d \Big)\ C(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)).$$

Proof:

To write the pyramid match in terms of counts of *unmatched* points in the implicit partial matching, we define the *directed distance* between two histograms formed from point sets $\mathbf{X}$ and $\mathbf{Y}$:

$$\begin{aligned} \mathcal{D}(H(\mathbf{X}), H(\mathbf{Y})) &= \sum_{j=1}^{r} \mathcal{D}_j \left(H(\mathbf{X})^{(j)}, H(\mathbf{Y})^{(j)} \right), \text{ where} \\ \mathcal{D}_j(a, b) &= \begin{cases} a - b, & \text{if } a > b \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

The directed distance $\mathcal{D}(H_i(\mathbf{X}), H_i(\mathbf{Y}))$ counts the number of unmatched points at resolution level i. Note that $\mathcal{D}(H_{-1}(\mathbf{X}), H_{-1}(\mathbf{Y})) = |\mathbf{X}|$ and $\mathcal{D}(H_{L-1}(\mathbf{X}), H_{L-1}(\mathbf{Y})) = 0$ by definition. The directed distance value decreases with i, as bins are increasing in size and more implicit matches are made. The change in the directed distance across levels is the decrease in the count of unmatched points from one level to the next, and therefore also serves to count the number of new matches formed at level i:

$$I(H_i(\mathbf{X}), H_i(\mathbf{Y})) - I(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})) = \mathcal{D}(H_{i-1}(\mathbf{X}), H_{i-1}(\mathbf{Y})) - \mathcal{D}(H_i(\mathbf{X}), H_i(\mathbf{Y})).$$

In terms of the directed distance, the pyramid match cost is

$$\begin{aligned}\mathcal{P}_\Delta(\mathbf{X},\mathbf{Y}) &= \sum_{i=0}^{L-1} d2^i\Big(\mathcal{D}(H_{i-1}(\mathbf{X}),H_{i-1}(\mathbf{Y})) - \mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\Big)\\ &= d\mathcal{D}(H_{-1}(\mathbf{X}),H_{-1}(\mathbf{Y})) + \sum_{i=0}^{L-2} d2^i\ \mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\\ &= d|\mathbf{X}| + \sum_{i=0}^{L-2} d2^i\ \mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\,.\end{aligned}$$

The expected value of the pyramid match cost is then

$$E\left[\mathcal{P}_\Delta(\mathbf{X},\mathbf{Y})\right] = d|\mathbf{X}| + \sum_{i=0}^{L-2} d2^i\ E\left[\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\right]. \tag{6}$$

The optimal partial matching $\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*) = \left\{(\mathbf{x}_1,\mathbf{y}_{\pi_1^*}),\ldots,(\mathbf{x}_m,\mathbf{y}_{\pi_m^*})\right\}$ implies a graph whose nodes are those points in the input sets $\mathbf{X}$ and $\mathbf{Y}$ that participate in the matching (i.e., all points in $\mathbf{X}$ and a subset from $\mathbf{Y}$), and whose edges connect the matched pairs $(\mathbf{x}_i,\mathbf{y}_{\pi_i})$, for $1 \le i \le |\mathbf{X}|$. Let n_j be the number of edges in this graph that have lengths in the range $[d2^{j-1}, d2^j)$. A bound on the optimal partial matching may then be expressed as:

$$\begin{aligned}\sum_j n_j\ d2^{j-1} &\le \mathcal{C}\left(\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*)\right), \text{ or}\\ \sum_j n_j\ d2^j &\le 2\ \mathcal{C}\left(\mathcal{M}(\mathbf{X},\mathbf{Y};\pi^*)\right),\end{aligned} \tag{7}$$

since for every j, all n_j edges must be of length at least $d2^{j-1}$.

The directed distance $\mathcal{D}$ counts at a given resolution how many points from set $\mathbf{X}$ are unmatched. Any edge in the optimal matching graph that is left "uncut" by the bin boundaries in histogram H_i contributes nothing to $\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))$. Any edge that is "cut" by the histogram contributes at most 1 to $\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))$. Therefore, the expected value of the directed distance at a given resolution is bounded by:

$$E\left[\mathcal{D}(H_i(\mathbf{X}),H_i(\mathbf{Y}))\right] \le \sum_j E\left[T_{ij}\right], \tag{8}$$

where T_{ij} is the number of edges in the optimal matching having lengths in the range $[d2^{j-1}, d2^j)$ that are cut by H_i.

The probability $\Pr(\text{cut}(\mathbf{x},\mathbf{y});i)$ that an edge for $(\mathbf{x},\mathbf{y})$ in the optimal matching is cut by the histogram grid H_i is bounded by $\frac{||\mathbf{x}-\mathbf{y}||_1}{2^i}$. This can be seen by considering the probability of a bin boundary cutting an edge in any one of the d dimensions, taking into account that the bin boundaries are shifted independently in each dimension. An edge with length $||\mathbf{x}-\mathbf{y}||_1 > d2^i$ is certainly cut by the histogram grid at H_i. An edge with length $||\mathbf{x}-\mathbf{y}||_1 \le d2^i$ may be cut in any dimension. In this case, the probability of a cut in dimension k for points $\mathbf{x} = [x_1,\ldots,x_d], \mathbf{y} = [y_1,\ldots,y_d]$ is $\frac{|x_k-y_k|}{2^i}$, for $1 \le k \le d$. The probability of a cut from H_i occurring in some dimension is then bounded by the sum of these probabilities:

$$\begin{aligned}\Pr(\text{cut}(\mathbf{x},\mathbf{y});i) &\le \sum_{k=1}^{d} \frac{|x_k-y_k|}{2^i}\\ &\le \frac{||\mathbf{x}-\mathbf{y}||_1}{2^i}.\end{aligned}$$

This provides an upper bound on the expected number of edges in the optimal matching that are cut at level i:

$$E\left[T_{ij}\right] \leq n_j \, \frac{d2^j}{2^i},$$

since by definition $d2^j$ is the maximum distance between any pair of matched points counted by n_j.

Now with Eqns. 7 and 8 we have a bound for the expected directed distance at a certain resolution level:

$$E\left[\mathcal{D}\left(H_i(\mathbf{X}), H_i(\mathbf{Y})\right)\right] \leq \frac{1}{2^i} \sum_j n_j \, d2^j \leq \frac{1}{2^i} \, 2 \, \mathcal{C}\left(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)\right).$$

Relating this to the expected value of the pyramid match cost in Eqn. 6, we have

$$\begin{aligned} E\left[\mathcal{P}_\Delta\left(\mathbf{X}, \mathbf{Y}\right)\right] &\leq d|\mathbf{X}| + \sum_{i=0}^{L-2} d2^i \left(\frac{1}{2^i} \, 2 \, \mathcal{C}\left(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)\right) \right) \\ &\leq d|\mathbf{X}| + 2d \, (L-1) \, \mathcal{C}\left(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)\right) \\ &\leq d|\mathbf{X}| + 2d \log D \, \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) \end{aligned} \tag{9}$$

since $L = \lceil \log_2 D \rceil + 1$.

As described in Section 3.1, points in sets $\mathbf{X}$ and $\mathbf{Y}$ are comprised of integer entries, insuring that the minimum inter-feature distance between unique points is 1. We also know that for the sake of measuring matching cost, $\mathbf{X}$ and $\mathbf{Y}$ are disjoint sets: $\mathbf{X} \cap \mathbf{Y} = \emptyset$; if there are any identical points in the two sets, we can consider them as discarded (in pairs) to produce strictly disjoint sets, since identical points contribute nothing to the matching cost. Since $\mathbf{X}, \mathbf{Y} \subseteq [D]^d$ and $\mathbf{X} \cap \mathbf{Y} = \emptyset$, we have $|\mathbf{X}| \leq \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*))$. That is, each edge in the optimal matching must have a length of at least 1, making the number of points in the smaller set a lower bound on the cost of the optimal matching for any two sets. With this bound and Eqn. 9, we have the following bound on the expected pyramid match cost error

$$\begin{aligned} E\left[\mathcal{P}_\Delta\left(\mathbf{X}, \mathbf{Y}\right)\right] &\leq d \, \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) + 2d \log D \, \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) \\ &\leq \left(2d \log D + d\right) \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)) \\ &\leq \left(C \cdot d \log D + d\right) \mathcal{C}(\mathcal{M}(\mathbf{X}, \mathbf{Y}; \pi^*)). \end{aligned}$$

7. Classification and Regression with the Pyramid Match

We train support vector machines and support vector regressors (SVRs) to perform classification and regression with the pyramid match kernel. An SVM or SVR is trained by specifying the matrix of kernel values between all pairs of training examples. The kernel's similarity values determine the examples' relative positions in an embedded space, and quadratic programming is used to find the optimal separating hyperplane or function between the two classes in this space. Because the pyramid match kernel is positive-definite we are guaranteed to find a unique optimal solution.

We have found that the pyramid match kernel can produce kernel matrices with dominant diagonals, particularly as the dimension of the features in the sets increases. The reason for this is that as the dimension of the points increases, there are a greater number of finer-resolution histogram levels in the pyramid where two input sets will have few shared bins. Once the quantization level is coarse enough, two sets will start having significant histogram intersection values. However, these

intersections will receive lower weights due to the $\frac{1}{d2^i}$ weighting scheme, which by construction accurately reflects the maximal distance between points falling into the same bin at level i. On the other hand, an input set compared against itself will result in large histogram intersection values at each level—specifically intersection values equal to the number of points in the set, which after normalization generates a diagonal entry of one.

The danger of having a kernel matrix diagonal that is significantly larger than the off-diagonal entries is that the examples appear nearly orthogonal in the feature space, in some cases causing an SVM or SVR to essentially "memorize" the data and impairing its sparsity and generalization ability (Shawe-Taylor and Cristianini, 2004). Nonetheless, we are able to work around this issue by modifying the initial kernel matrix in such a way that reduces its dynamic range, while preserving its positive-definiteness. We use the functional calculus transformation suggested in Weston et al. (2002): a subpolynomial kernel is applied to the original kernel values, followed by an empirical kernel mapping that embeds the distance measure into a feature space. Thus, when necessary to reduce diagonal dominance, first kernel values $\mathbf{K}_{ij}$ generated by $\mathcal{P}_\Delta$ are updated to $\mathbf{K}_{ij} \leftarrow {\mathbf{K}_{ij}}^p$, $0 < p < 1$. Then the kernel matrix $\mathbf{K}$ is replaced with $\mathbf{K}\mathbf{K}^{\mathrm{T}}$ to obtain the empirical feature map $\Phi_e(\mathbf{y}) = [K(\mathbf{y},\mathbf{x}_1),\ldots,K(\mathbf{y},\mathbf{x}_N)]^{\mathrm{T}}$ for N training examples. As in Weston et al. (2002), the parameter p is chosen with cross-validation. This post-processing of the kernel matrix is not always necessary; both the dimension of the points as well as the specific structure of a given data set will determine how large the initial kernel matrix diagonal is.

8. Empirical Quality of Approximate Partial Matchings

The approximation bounds we can show are actually significantly weaker than what we observe in practice. In this section, we empirically evaluate the approximation quality of the pyramid match, and make a direct comparison between our partial matching approximation and the L_1 embedding of Indyk and Thaper (2003).

We conducted experiments to evaluate how close the correspondences implicitly measured by the pyramid match are to the true optimal correspondences—the matching that results in the minimal summed cost between corresponding points. In order to work with realistic data but still have control over the sizes of the sets and the amount of clutter features, we established synthetic "category" models. Each model is comprised of some fixed number m' of parts, and each part has a Gaussian model that generates its d-dimensional appearance vector (in the spirit of the "constellation model" used by Fergus et al., 2003, and others). Given these category models, we can then add clutter features, adjust noise parameters, and so on, simulating in a controlled manner the variations that occur with the sets of image patches extracted from an actual object. The appearance of the clutter features is determined by selecting a random vector from a uniform distribution on the same range of values as the model features.

We generated two data sets, one with equally-sized sets, and one with variable-sized sets. Every point set was drawn from one of two synthetic category models, with $m' = 35$ and $d = 2$. For the first data set, 50 point sets containing only the m' model features were sampled from both of the two category models, for a total of 100 equally-sized point sets. For the other data set, the model feature sets were merged with a randomly generated number C of "extra" clutter features, for a total of 100 point sets with $m' + C$ features each, with C selected uniformly at random from $[0, 100]$. We compared the pyramid match's outputs to those produced by the optimal partial matching obtained via a linear programming solution to the transportation problem (Rubner et al., 2000), as well as

those produced by an L_1 approximation (Indyk and Thaper, 2003). For both of the data sets, we computed the pairwise set-to-set distances using each of these three measures.

If an approximate measure is approximating the optimal matching well, we should find the ranking induced by that approximation to be highly correlated with the ranking produced by the optimal matching for the same data. In other words, the point sets should be sorted similarly by either method. We can display results in two ways to evaluate if this is true: 1) by plotting the actual costs computed by the optimal and approximate method, and 2) by plotting the rankings induced by the optimal and approximate method. Spearman's rank correlation coefficient R provides a good quantitative measure to evaluate the ranking consistency:

$$R = 1 - \frac{6\sum_{i=1}^{N}(i-\hat{r}(i))^2}{N(N^2-1)},$$

where i is the rank value in the true order and $\hat{r}(i)$ is the corresponding rank assigned in the approximate ordering, for each of the N corresponding ordinal values assigned by the two measures.

Figure 4 displays both types of plots for the two data sets: the top row (a) displays plots corresponding to the data set with equally-sized sets, that is, for the bijective matching problem, while the bottom row (b) displays plots corresponding to the data set with variable-sized sets, that is, for the partial matching problem.

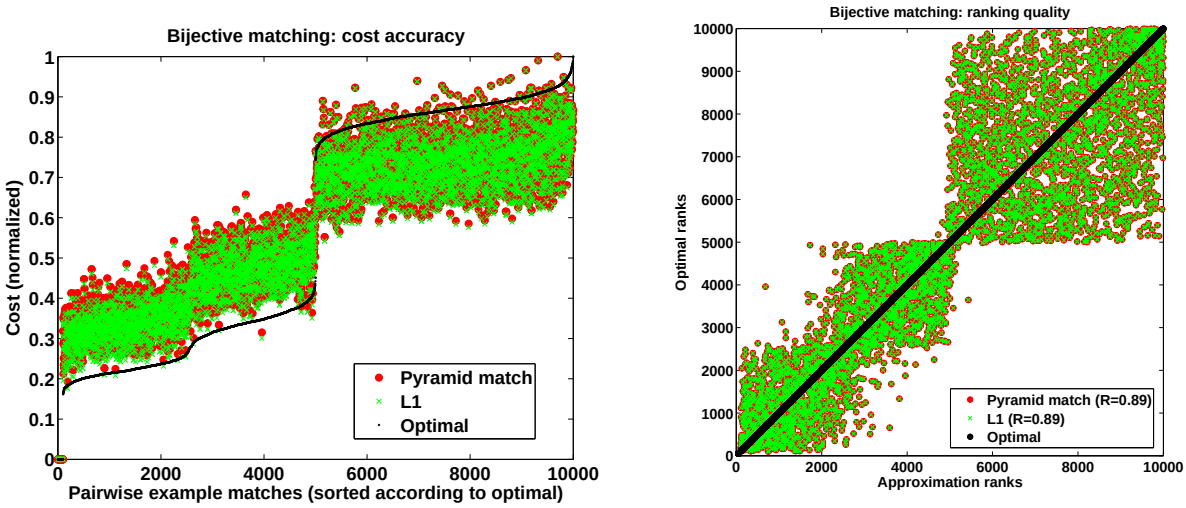
The two plots in the lefthand column show the normalized output costs from 10,000 pairwise set-to-set comparisons computed by the optimal matching (black), the pyramid match with the number of random shifts $T = 3$ (red circles), and the L_1 approximation, also with $T = 3$ (green x's). Note that in these figures we plot cost (so the pyramid match weights are set to $w_i = d2^i$), and for display purposes the costs have been normalized by the maximum cost produced for each measure to produce values between $[0,1]$. The cost values are sorted according to the optimal measure's magnitudes for visualization purposes. The raw values produced by the approximate measures will always overestimate the cost; normalizing by the maximum cost value simply allows us to view them against the optimal measure on the same scale.

The two plots in the righthand column display the rankings for each approximation plotted against the optimal rankings. The black diagonals in these plots denote the optimal performance, where the approximate rankings would be identical to the optimal ones. The R values displayed in the legends refer to the Spearman rank correlation scores for each approximation in that experiment; higher Spearman correlations have points clustered more tightly along this diagonal.

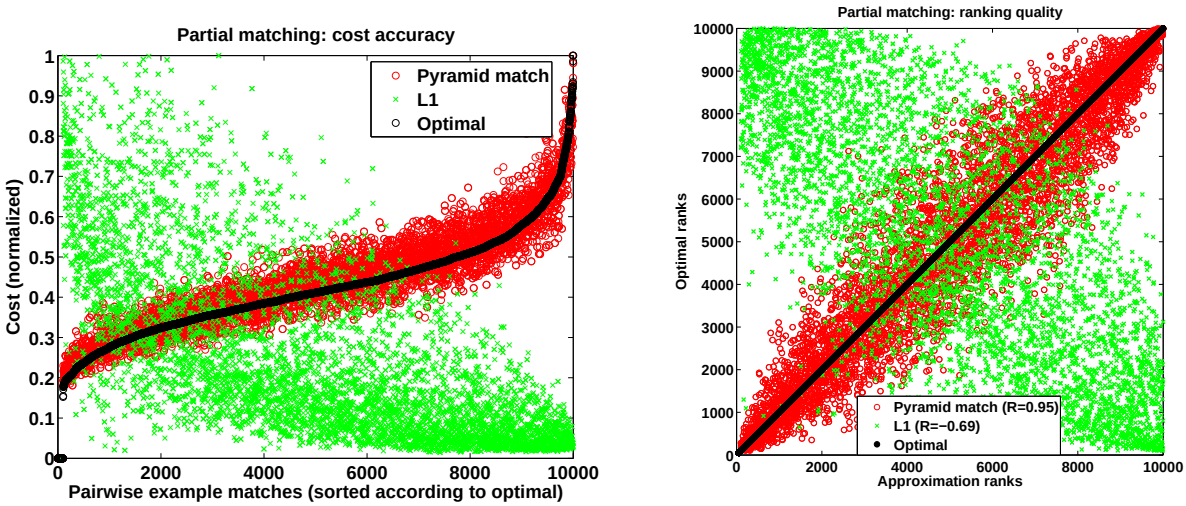
Both approximations compute equivalent costs and rankings for the bijective matching case, as indicated by Figure 4 (a). This is an expected outcome, since the L_1 distance over histograms is directly related to histogram intersection *if* those histograms have equal masses (Swain and Ballard, 1991), as they do for the bijective matching test:

$$I(H(\mathbf{Y}),H(\mathbf{Z})) = m - \frac{1}{2}||H(\mathbf{Y}) - H(\mathbf{Z})||_{L_1}, \text{ if } m = |\mathbf{Y}| = |\mathbf{Z}|.$$

The structure along the diagonal in the top right plot reflects the fact that two types (categories) of point sets were present in the data, causing items of the same category to block together when they are sorted by matching cost. The square-shaped cluster in the upper right portions of the plots show that while the approximate measures do not distinguish between all examples within a category precisely the way the optimal measure does, they do both consider all items within one category to be most distant from an example drawn from the other category. Similarly, there are discontinuities



(a) Bijective matching



(b) Partial matching

Figure 4: Pyramid match and L_1 embedding comparison on (a) bijective matchings with equally-sized sets and (b) partial matchings with variably-sized sets. When all input sets are the same size, the pyramid match and the L_1 embedding give equivalent results. However, the pyramid match also approximates the optimal correspond ences for input sets of unequal cardinalities and allows partial matches. (This figure is best viewed in color.)

Figure 5: Example images from the ETH-80 objects database. Five images from each of the eight object classes (apple, cow, dog, pear, car, cup, horse, and tomato) are shown here.

in the left plot of part (a) due to distance clusters caused by drawing point sets from two distinct category models.

However, for the partial matching case (Figure 4 (b)), the L_1 approximation fails because it can handle only sets with equal cardinalities and requires all points to match to something. In contrast, the pyramid match can also approximate the partial matching for the unequal cardinality case: its matchings continue to follow the optimal matching's trend since it does not penalize outliers, as evidenced by the clustered points along the diagonal in the bottom right ranking quality plot. We have performed this same experiment using data generated from a uniform random point model, and the outcome was similar.

9. Discriminative Classification using Sets of Local Features

In this section we report on object recognition experiments using SVMs and provide baseline comparisons to other methods. We use the SVM implementation provided in the LIBSVM library (Chang and Lin, 2001) and train one-versus-all classifiers in order to perform multi-class classification.

Local affine- or scale-invariant feature descriptors extracted from a sparse set of interest points in an image have been shown to be an effective, compact representation (e.g., Lowe 2004; Mikolajczyk and Schmid 2001). This is a good context in which to test the pyramid match kernel, since such local features have no inherent ordering, and it is expected that the number of features will vary across examples. Given two sets of local image features, the pyramid match kernel (PMK) value reflects how well the image parts match under a one-to-one correspondence. Since the matching is partial, not all parts must have a match for the similarity to be strong.

In the following we experiment with two publicly available databases and demonstrate that our method achieves better or comparable object recognition performance at a significantly lower computational cost than other state-of-the-art approaches. All pyramid match run-times reported below include the time needed to compute both the pyramids and the weighted intersections, and were measured on 2.4 GHz processors with 2 GB of memory.

9.1 ETH-80 Data Set

A performance evaluation given by Eichhorn and Chapelle (2004) compares the set kernels developed by Kondor and Jebara (2003), Wolf and Shashua (2003), and Wallraven et al. (2003) in the context of an object categorization task using images from the publicly available ETH-80 database.[4] The experiment uses eight object classes, with 10 unique objects and five widely separated views of each, for a total of 400 images (see Figure 5). A Harris detector is used to find interest points in each image, and various local descriptors (SIFT features, Lowe 2004; JETs, Schmid and Mohr 1997; and raw image patches) are used to compose the feature sets. A one-versus-all SVM classifier is trained for each kernel type, and performance is measured via cross-validation, where all five views of an object are held out at once. Since no instances of a test object are ever present in the training set, this is a categorization task (as opposed to recognition of the same specific object).

The experiments show the polynomial-time "match kernel" (Wallraven et al., 2003) and Bhattacharyya kernel (Kondor and Jebara, 2003) performing best, with a classification rate of 74% using on average 40 SIFT features per image (Eichhorn and Chapelle, 2004). Using 120 interest points, the Bhattacharyya kernel achieves 85% accuracy. However, the study also concluded that the cubic complexity of the method made it impractical to use the desired number of features.

We evaluated the pyramid match kernel on this same subset of the ETH-80 database under the conditions provided in Eichhorn and Chapelle (2004), and it achieved a recognition rate of 83% using PCA-SIFT features (Ke and Sukthankar, 2004) from all Harris-detected interest points (averages 153 points per image) and $T = 8$. Restricting the input sets to an average of 40 interest points yields a recognition rate of 73%. Thus the pyramid match kernel (PMK) performs comparably to the others at their best for this data set, but is much more efficient than those tested above, requiring time only linear in the number of features.

In fact, the ability of a kernel to handle large numbers of features can be critical to its success. An interest operator may be tuned to select only the most salient features, but in our experiments we found that recognition rates always benefited from having larger numbers of features per image with which to judge similarity. The plots in Figure 6 depict the run-time versus recognition accuracy of the pyramid match kernel as compared to the match kernel (Wallraven et al., 2003), which has $O(dm^2)$ complexity. The match kernel computes kernel values by averaging over the distances between every point in one set and its nearest point in the other set. These are results from our own implementation of the match kernel. Each point in the figure represents one experiment, and both kernels were run with the exact same PCA-SIFT feature sets. The saliency threshold of the Harris interest operator was adjusted to generate varying numbers of features, thus trading off accuracy versus run-time. With no filtering of the Harris detector output (i.e., with a threshold of 0), there is on average 153 features per set. To extract an average of 256 features per set we used no interest operator and sampled features densely and uniformly from the images. Computing a kernel matrix for the same data with the two kernels yields nearly identical accuracy (left plot), but the pyramid match is significantly faster (center plot). Allowing the same amount of training time for both methods, the pyramid match produces much better recognition results (right plot).

9.2 Caltech-101 Data Set

We also tested our method with a challenging database of 101 object categories developed at Caltech, called the Caltech-101 (Fei-Fei et al., 2004). The creators of this database obtained the images

4. http://www.vision.ethz.ch/projects/categorization/.

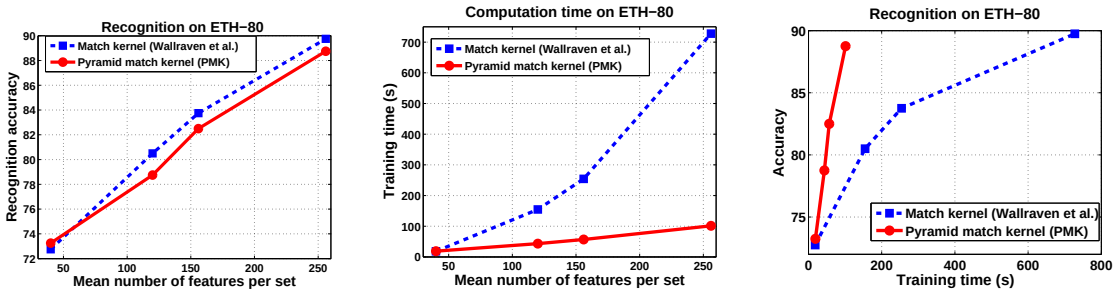


Figure 6: Both the pyramid match kernel and the match kernel of Wallraven et al. yield comparable recognition accuracy for input sets of the same size (left plot). Both benefit from using richer (larger) image descriptions. However, while the time required to learn object categories with the quadratic-time match kernel grows quickly with the input size, the time required by the linear-time pyramid match remains very efficient (center plot). Allowing the same run-time, the pyramid match kernel produces better recognition rates than the match kernel (right plot). Recognition accuracy is measured as the mean rate of correct predictions per class.

using Google Image Search, and many of the images contain a significant amount of intra-class appearance variation (see Figure 7). The Caltech-101 is currently the largest benchmark recognition data set (in terms of number of categories) available, and as such it has been the subject of many recent recognition experiments in the field. There are 8677 images in the data set, with between 31 to 800 images for each of the 101 categories.

For this data set, the pyramid match operated on sets of SIFT features projected to $d = 10$ dimensions using PCA, with each appearance descriptor concatenated with its corresponding positional feature (image position normalized by image dimensions). The features were extracted on a uniform grid at every 8 pixels in the images, that is, no interest operator was applied. The regions extracted for the SIFT descriptor were about 16 pixels in diameter, and each set had on average $m = 1140$ features. We trained the algorithm using unsegmented images. Since the pyramid match seeks a strong correspondence with some subset of the images' features, it explicitly accounts for unsegmented, cluttered data. Classification was again done with a one-vs-all SVM, and we set the number of grid shifts $T = 1$, and summed over kernels corresponding to pyramids with finest side lengths of 5,7, and 9. Note that some images were rotated by the creators of the database, causing some triangular artifacts in the corners of some images; this is how the images are provided to any user.

As a baseline, we also experimented with the bag-of-words representation and an RBF kernel: $K(\mathbf{x},\mathbf{y}) = \exp(-||\mathbf{x}-\mathbf{y}||/\sigma)$ for bag-of-words histograms $\mathbf{x}$ and $\mathbf{y}$. We used hierarchical k-means to cluster a corpus of two million example features from the Caltech-101 data set to form each vocabulary, and we experimented with both L_2 and the Mahalanobis distance as the basis for the clustering; classification results were similar with either distance. Generating a vocabulary took about five hours, and computing a kernel matrix required about one half hour. We experimented with a large number of parameters, and the dotted red line in Figure 8 (a) reflects the very best results we could obtain over any parameter setting for this technique. Specifically, the number of

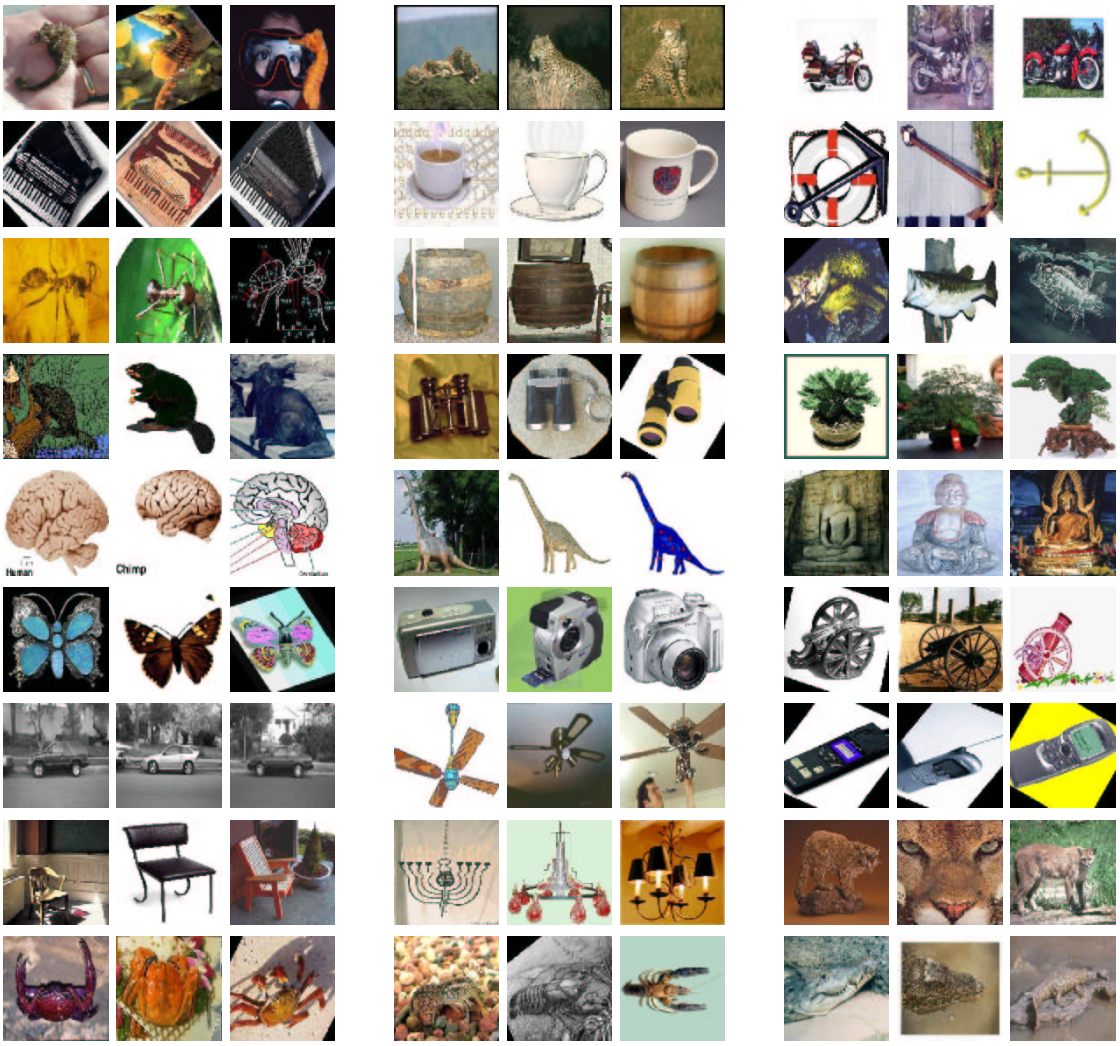


Figure 7: Example images from the Caltech-101 database. Three images are shown for each of 27 of the 101 categories.

feature prototypes ("words") was tested at values ranging from 1000 to 100,000 (2200 was best); the kernel's σ parameter was tested from values of 500 to 10,000 (1000 was best).

Figure 8 (a) shows the multi-class category recognition results using the PMK (red solid line) alongside the corresponding results for a bag-of-words baseline approach (blue dotted line). The recognition scores are given for varying numbers of training set sizes, ranging from one to 30 examples per class. For each training set size, the mean and standard deviation of the pyramid match accuracy over 10 runs is shown, where for each run we randomly select the training examples and use all the remaining database images as test examples. All recognition rates have been normalized according to the number of novel test images per class; that is, the mean recognition rate per class is the average normalized score for all 101 categories. Using only one training example per class, the pyramid match achieves an average recognition rate per class of 18%. Note that chance per-

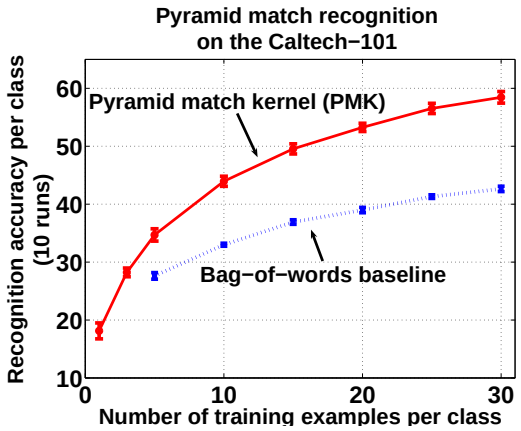
formance with any number of training examples would be just 1%. Using 15 examples (a standard point of comparison), the pyramid match achieves an average recognition rate per class of 50%. Experiments comparing the recognition accuracy of the pyramid match kernel to an optimal partial matching kernel reveal that very little loss in accuracy is traded for speedup factors of several orders of magnitude (Grauman, 2006, , pages 118-121).

Over all the parameter configurations we tested, the best performance we could achieve with the bag-of-words on this data set is substantially poorer than that of the pyramid match, as seen in Figure 8 (a). In addition, the bag-of-word approach's accuracy was fairly sensitive to the parameter settings; for instance, over all the parameters we tested, the mean accuracy for 15 training examples per class varied from 28% to 37%. This indicates the difficulty of establishing an optimal flat quantization of the feature space for recognition, and also suggests that the partial matching ability of the pyramid match is beneficial for tolerating outlier features without skewing the matching cost.

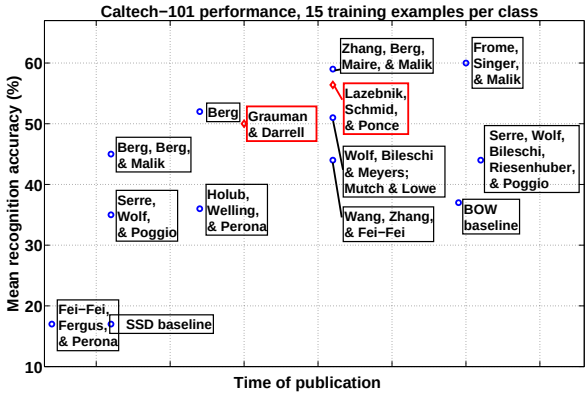
Figure 8 (b) shows all published results on this data set as a function of time since the data was released, including the PMK and results published more recently by other authors (Holub et al. 2005b; Serre et al. 2007; Wolf et al. 2006; Wang et al. 2006; Berg et al. 2005; Berg 2005; Fei-Fei et al. 2004; Mutch and Lowe 2006; Lazebnik et al. 2006; Zhang et al. 2006; Frome et al. 2007). From this comparison we can see that even with its extreme computational efficiency, the pyramid match kernel achieves results that are competitive with the state-of-the-art. We have previously obtained 50% accuracy on average (0.9% standard deviation) when using the standard 15 training examples per category (Grauman and Darrell, 2006). In addition, the PMK with sets of spatial features yields very good accuracy on this data set: 56.4% for 15 training examples per class (Lazebnik et al., 2006). These results are based on a special case of the pyramid match, where multiple pyramid match kernels computed on spatial features are summed over a number of quantized appearance feature channels. This is among the very best accuracy rates reported to-date on the data set, and about three percentage points below the most accurate result of 60% obtained recently by Frome et al. (2007), which compares images using a discriminative local distance function.

Figure 8 (c) shows the recognition accuracy of various methods as a function of computation time, measured in terms of the time required to train a classifier (left plot) and the time required to classify a novel example (right plot). We collected these time estimates directly from the authors, and every response we received is plotted here. (So not every method represented in (b) is represented in (c) due to a lack of information on computation times.) For both plots in part (c), computation time is measured *excluding* the time required to extract image features for all methods. For display purposes, in the left plot, we plot the square root of the training time; nearest-neighbor classifiers such as the method of Berg et al. require no training time beyond feature extraction. In the right plot, we plot the log of the classification time estimates to reflect the relative complexities in terms of orders of magnitude. This figure illustrates the PMK's clear practical cost advantages. Pyramid matching any two examples requires just 0.0001 *s* on this data set, and classifying a novel example requires a fraction of a second; in contrast, many other techniques require on the order of a minute to classify a single example, and yield varying accuracies relative to the PMK.

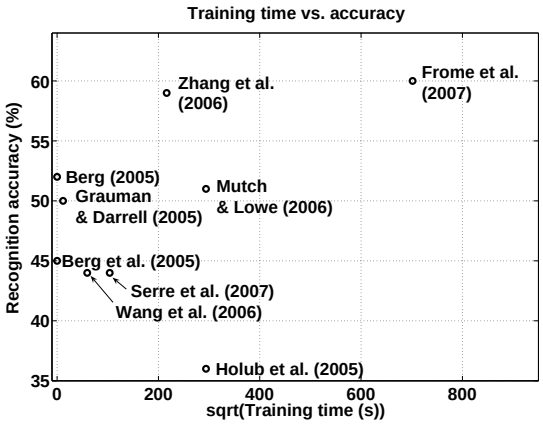
Pyramid match comparisons require only $O(mL)$ time for the result of 50% accuracy. For the spatial pyramid match result of 56.4% accuracy obtained by Lazebnik et al. (2006) there is an additional one-time $O(mk)$ cost of assigning each of a set's features to one of the k pre-established quantized appearance features; however, once a point set has been mapped to its prototypes, they need not be re-computed to match against additional sets.

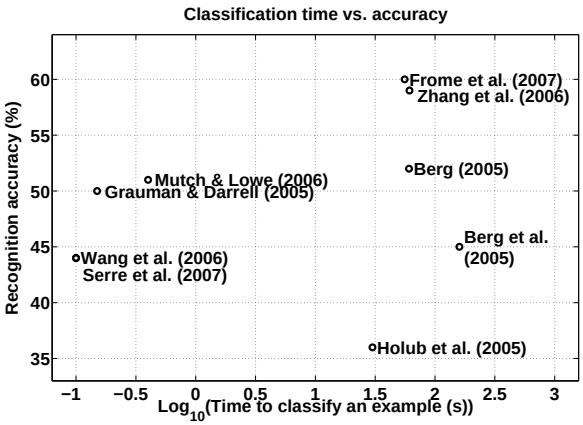


(a) PMK vs. BOW categorization results



(b) Comparison of all published results





(c) Accuracy vs. time complexity

Figure 8: Recognition results on the Caltech-101 data set. Plot (a) shows the mean and standard deviation for the recognition accuracy using the pyramid match kernel (red solid line) and a bag-of-words baseline (blue dotted line) when given varying numbers of training examples per class, over 10 runs with randomly selected training examples. These are recognition rates that have been normalized according to the number of test examples per class. Plot (b) shows all published results on the same Caltech-101 data set over time since the database's release. Each recognition rate on this plot represents the mean accuracy per class when using 15 training examples per class. Grauman & Darrell refers to our PMK results using appearance descriptors, and Lazebnik et al. report results using a PMK applied to 2D spatial features in each channel of quantized appearance. The two plots in (c) show the accuracy of various methods as a function of their computational costs, in terms of average training (left, square root scale) and testing times (right, log scale), again when each method uses 15 training examples per class. All times exclude the cost of computing the various image features employed.

10. Learning a Function over Sets of Features

In the following experiments we demonstrate the pyramid match applied to two regression problems: time of publication inference from a collection of research papers, and articulated pose estimation from monocular silhouettes. We again use the SVR implementation of Chang and Lin (2001). In these experiments we use an ε-insensitive loss function in order to obtain solutions that are defined in terms of a small subset of the training examples, and which provide good generalization ability. For all experiments, the SVR parameters C and ε were chosen using cross-validation.

10.1 Estimating a Document's Time of Publication

We have applied our method to learn a function that maps a bag of local latent semantic features extracted from a research paper (written in a specific field) to an estimate of the paper's year of publication. While much work has been done using bag-of-words representations and latent semantic concept models for text classification and information retrieval, we are not aware of previous work showing *bags of local semantic features*, nor demonstrating direct regression on documents to estimate their publication times.

The bag-of-words model is a widely used representation in text processing in which each document is represented by a vector giving the frequencies of words that occur in it, and it has been used in kernel methods (Shawe-Taylor and Cristianini, 2004). The well-known limitation of such a model, however, is that it encodes nothing about the semantic relationships between words. Each word is treated as an isolated entity, ignoring the fact that distinct words may share degrees of semantic equivalency (e.g., synonyms), or may have different connotations in different contexts (e.g., homonymy). Researchers have therefore adopted latent semantic indexing (LSI) (Deerwester et al., 1990) to instill semantic cues into the basic bag-of-words representation. LSI projects data onto a subspace determined using singular value decomposition (SVD) to represent document-term frequency vectors in a lower-dimensional space where semantic relations are better preserved. Generally the subspace for LSI is learned from document vectors that give the frequency of occurrence for each given word (e.g., Cristianini et al., 2002), which means each document is mapped to a point in a "semantic space" where documents with correlated word frequencies are related.

However, requiring a document to map to a single point in this space assumes that inputs have no clutter (e.g., extra words caused from OCR errors, or text inserted from a webpage ad), and that each document can be represented as a linear combination of the document-level concepts recovered by LSI. Instead, we propose treating documents as bags of *word meanings* by learning the subspace for LSI from *term* vectors, which record the frequency with which a word occurs in each given document in a training corpus. Each document is then a bag of local semantic features, and two documents are compared with the partial matching implied by the pyramid match kernel, that is, in terms of how well (some subset) of the LSI term-space projections can be put into correspondence. Note that standard kernels (e.g., linear, RBF, polynomial) cannot be used with the bag of word meanings representation, since it represents each word with a real-valued vector. Additionally, our method makes it possible to learn a latent semantic space from narrower contexts than entire documents (e.g., paragraphs or sentences) and then represent documents by their component features in this space.

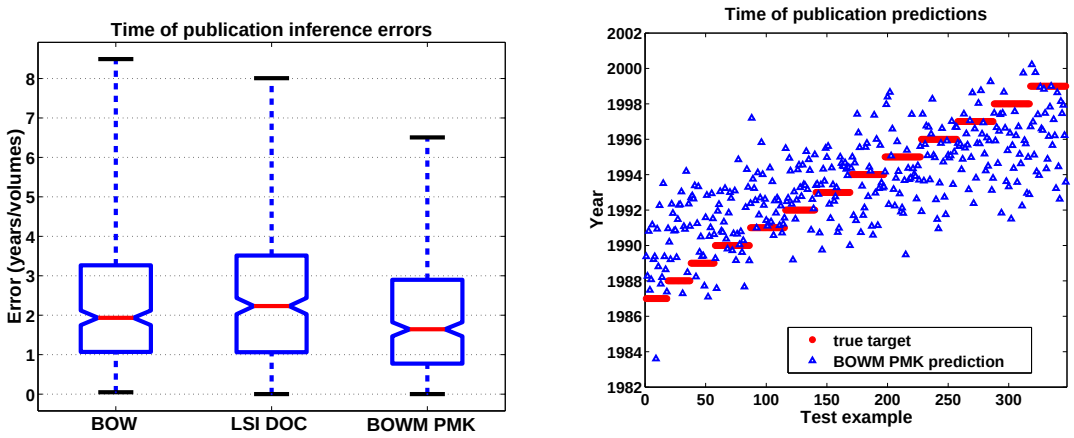


Figure 9: Inferring the time of publication for papers from 13 volumes of NIPS proceedings. Boxplots (left) compare errors produced by three methods with a support vector regressor: bag-of-words (BOW) and latent semantic document-vectors (LSI DOC) with linear kernels, and "bag of word meanings" with the pyramid match kernel (BOWM PMK). Lines in center of boxes denote median value, top and bottom of boxes denote upper and lower quartile values, dashed lines show the extent of the rest of the errors. The plot on the right shows the true targets and corresponding predictions made by the pyramid match method (BOWM PMK).

We have experimented with a database containing 13 volumes of Neural Information Processing Systems (NIPS) proceedings—a total of 1,740 documents, available online.[5] For each paper, we extracted the text from the abstract up to (but not including) the references section. While authors' names and bibliography information would likely be good indicators of publication time, they were excluded from the bags of features because we want our method to learn a function indicating topic trends over time, as opposed to a look-up table of dates and names. We applied standard steps to pre-process the text data. Suffixes were removed from words using the Porter stemmer (Porter, 1980), and the WordNet set of stop-list words were removed. Finally, co-occurrence matrices were weighted using term frequency-inverse document frequency (*tf-idf*) to normalize for text lengths and distill the most informative words.

Figure 9 shows results for regressing directly on the year of publication for a NIPS paper using the classic bag-of-words approach (BOW), a standard approach applying LSI at the document level (LSI DOC) (Cristianini et al., 2002), and our method with bags of word meanings (BOWM PMK). All methods were trained with the same randomly selected subset of the data (1393 examples) and tested with the remaining 347 examples. Our approach with bags of word meanings performs the best, with a median error of 1.6 years. The PMK values took on average 0.002 seconds to compute. Using a paired difference T-test with $\alpha = 0.01$, we found the difference in performance between our approach and the two existing methods to be statistically significant.

5. http://www.cs.toronto.edu/ roweis/data.html.

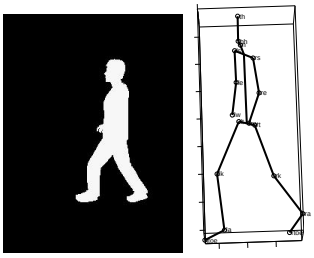


Figure 10: A training example generated with graphics software is composed of a silhouette and its corresponding 3-D pose, as represented by the 3-D positions of 15 joint positions.

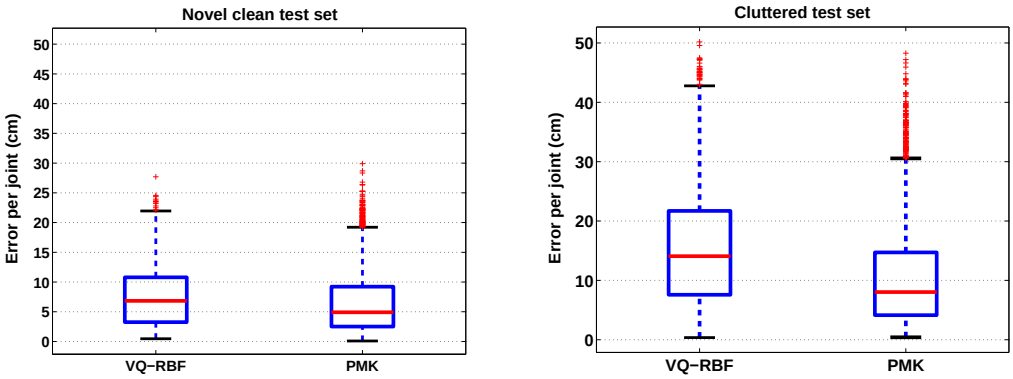
10.2 Inferring 3-D Pose from Shape Features

In this set of experiments, we use regression with the pyramid match kernel to directly learn the mapping between monocular silhouette images of humans and the corresponding articulated 3-D body pose. Many vision researchers have addressed the difficult problem of articulated pose estimation; recent approaches have attempted to directly learn the relationship between images and 3-D pose (Agarwal and Triggs, 2004; Shakhnarovich et al., 2003; Grauman et al., 2003). Like these techniques, we learn a function that maps observable image features to 3-D poses.

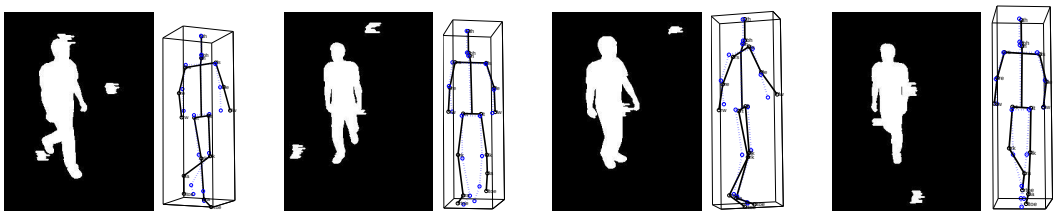
However, whereas ordered, fixed-length feature sets are required in the methods of Grauman et al. (2003) and Shakhnarovich et al. (2003) (i.e., points are extracted in sequence around the contour, or features are taken from fixed image windows), our method accepts unordered features and inputs that may vary in size. This is a critical difference: images will naturally have varying numbers of features (due to occlusions, clutter, translations, shape deformations, viewpoint changes, etc.), and a robust global ordering among features within a single view may not be possible in the presence of viewpoint and pose variations. In the pose estimation method of Agarwal and Triggs (2004) a bag-of-words representation is used: local features are mapped to pre-established prototypes, and every image is represented by a frequency vector counting the number of times each prototype occurred. A relevance vector machine is then trained using these vectors with a Gaussian kernel. While this representation allows unordered features, it can be sensitive to clutter, as we will show below.

As a training set, we used 3,040 images of realistic synthetic images of pedestrians generated using the graphics package POSER. Each image was rendered from a humanoid model with randomly adjusted anatomical shape parameters walking in a randomly selected direction. For each image, both the silhouette and the 3-D locations of 15 landmarks on the model's skeleton corresponding to selected anatomical joints were recorded (see Figure 10). Regressors are trained with silhouette inputs and produce 3-D joint position outputs. Once regressors have been trained with the synthetic data, we can use them to perform regression with either additional synthetic examples (for which we have ground truth poses), or with real image data (for which we can only evaluate predicted poses qualitatively).

We represent each silhouette with a set of local contour descriptors. At each contour point, we extract a shape context histogram (Belongie et al., 2002), which bins nearby edge pixels into a log-polar histogram, thereby encoding local shape. For each shape context histogram, we used 12 angular and five radial bins with a fixed scale to capture only local points relative to each edge point. To form a more compact descriptor, we used low-dimensional PCA projections ($d = 5$) of



(a) Pose estimation errors



(b) Noisy synthetic test examples

Figure 11: Pose inference results. The top row (a) gives a quantitative evaluation of performance on synthetic data with ground truth. The boxplots compare the error distributions for the pyramid match kernel (PMK) and an RBF kernel over prototype frequency vectors (VQ-RBF). Errors are measured by the distance between the 15 true and inferred joint positions. Results for two test sets are shown: a set of novel, clean silhouettes (left plot), and a set with randomly generated clutter or extra foreground blobs (right plot). The bottom row (b) shows example poses inferred by our method from synthetic cluttered silhouettes. In each, the true pose (solid black) is overlayed with the estimate (dotted blue). These examples contain average case errors.

the initial 60-D shape context histogram features. Thus each silhouette shape is represented by an unordered set of shape context subspace features, and each set varies in size due to the varying number of contour points per image. Note that although this representation does not contain explicit spatial constraints, the overlap between neighboring shape context histograms provides an implicit encoding of how the features should be related spatially.

The number of contour points (and thus features) per image varied from $m = 405$ to $m = 878$. The multi-resolution histograms used by the pyramid match contained ten resolution levels, as determined by the diameter of the features in the training examples, and each contained on average 2644 non-empty bins. Computing a PMK value required about 0.002 seconds. For each dimension of the pose targets, we trained an ε-insensitive SVR using the pyramid match kernel matrix between the training examples. Each SVR had on average 308 support vectors.

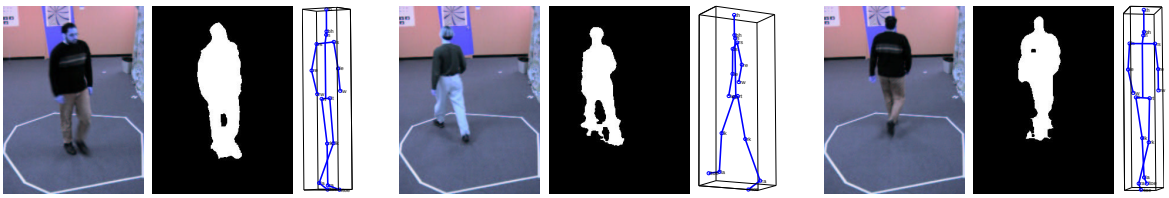


Figure 12: Pose inference on real images.

As a baseline, we also implemented a method that uses frequency vectors over feature prototypes to represent each image (using Agarwal and Triggs, 2004, as a guideline). To obtain bag-of-words histograms, vector quantization (VQ) is performed on the shape context subspace features found in the training set to establish a set of 100 prototype features. Then all of the features detected in a new image are mapped to a 1-D frequency histogram over these prototypes using soft voting with Gaussian weights. A Gaussian RBF kernel is then applied, with γ chosen based on the maximum inter-feature distance. In the following we will refer to this baseline as VQ-RBF.

For a novel test set of 300 POSER-generated silhouettes, the pose inferred by our method had a median error of 4.9 cm per joint position. For the same test set, VQ-RBF obtained a slightly worse median error of 6.8 cm (see Figure 11). Using a paired difference T-test with $\alpha = 0.001$, we found the difference in performance between the two methods to be statistically significant.

The silhouette contours produced with POSER are of course very "clean", that is, the shapes are perfectly segmented since they were formed by directly projecting the CG body. While it is reasonable to train a model with this well-segmented data, we can expect real-world examples to exhibit poorer foreground-background segmentations due to occlusions, clutter, shadows, or backgrounds that look similar to the foreground object. Therefore, we altered the silhouettes for a separate test set of 300 examples to introduce clutter and occlusions; starting with a POSER silhouette, we generated one to eight random blob regions in the image for which the foreground/background labels were swapped. The blobs' positions, sizes, and shapes were generated randomly. The result is a test set that attempts to mimic real-world occlusions and clutter, yielding imperfect contours for which estimating pose is more challenging (see Figure 11).

On the cluttered test set, our method inferred poses with a median error of 8.0 cm per joint, while VQ-RBF had a median error of 14.1 cm (see Figure 11). Again, using a paired difference T-test, we found the difference in performance to be statistically significant: with 99.99% confidence, the pyramid match yields average errors that are smaller than those of VQ-RBF by amounts between 4.5 and 5.2 cm per joint.

This experiment demonstrates the pyramid match kernel's robustness to superfluous features in an input. The blobs added to the cluttered test set introduced extra contour features to the silhouettes, and they altered parts of the true contour in cases where they overlapped with the real silhouette. The VQ-RBF distance over prototype frequency vectors (bags-of-words) essentially penalizes any features that have no "match" in a support vector training example's set of features. In contrast, the pyramid match's partial matching rewards similarity only for the features placed into correspondence, and ignores extraneous clutter features. This is an important advantage for many vision applications, where segmentation errors, viewpoint changes, or image noise will often affect which features (and how many) are detected.

Finally, we applied our method to a test set of real images of various subjects walking through a scene. A basic background subtraction method was used, which resulted in rough segmentations; body parts are frequently truncated in the silhouettes where the background is not highly textured, or else parts are inaccurately distended due to common segmentation problems from shadows. Ground truth poses are not available for this test set, but Figure 12 shows some example output poses inferred by our method.

11. Conclusions

We have developed a new efficient kernel function that handles unordered sets of features, and we have shown its suitability for both discriminative classification and regression. The pyramid match kernel approximates the optimal partial matching by computing a weighted intersection over multi-resolution histograms, and it requires time only linear in the number of features. The kernel is robust to clutter since it does not penalize the presence of extra features, it respects the co-occurrence statistics inherent in the input sets, and it is provably positive-definite. We have applied our kernel to a variety of tasks—object recognition, pose estimation, and time of publication inference—and have demonstrated our method's advantages over other state-of-the-art techniques. Source code for the pyramid match kernel is available at `http://people.csail.mit.edu/jjl/libpmk/`.

Recently we have developed a method to generate data-dependent pyramid partitions, which yield more accurate matching results in high-dimensional feature spaces (Grauman and Darrell, 2007). In ongoing work we are considering alternative weighting schemes on the bins for the pyramid match, and developing methods for sub-linear time approximate similarity search over partial correspondences.

Acknowledgments

We would like to thank Piotr Indyk, Regina Barzilay, Michael Collins, and John Lee for helpful discussions, and Michael Collins for suggesting the application of estimating publication times. We also thank John Lee for sharing his bag-of-words results on the Caltech-101 database, and the JMLR reviewers for their helpful comments.

References

A. Agarwal and B. Triggs. 3D human pose from silhouettes by relevance vector regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Washington, DC, June 2004.

P. Anandan. *Measuring Visual Motion from Image Sequences*. PhD thesis, University of Massachusetts, Amherst, 1987.

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24):509–522, April 2002.

A. Berg. *Shape Matching and Object Recognition*. PhD thesis, U.C. Berkeley, Computer Science Division, Berkeley, CA, December 2005.

A. Berg, T. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 2005.

S. Boughhorbel, J-P. Tarel, and F. Fleuret. Non-Mercer kernels for SVM object recognition. In *British Machine Vision Conference*, London, UK, September 2004.

S. Boughorbel, J. Tarel, and N. Boujemaa. The intermediate matching kernel for image local features. In *International Joint Conference on Neural Networks*, Montreal, Canada, August 2005.

P. Burt and E. Adelson. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, COM-31,4:532–540, 1983.

C. Chang and C. Lin. *LIBSVM: a library for SVMs*, 2001.

O. Chapelle, P. Haffner, and V. Vapnik. SVMs for histogram-based image classification. *Transactions on Neural Networks*, 10(5), September 1999.

T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *Journal of Intelligent Information Systems*, 18(2/3):127–152, 2002.

G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. In *Proceedings of European Conference on Computer Vision*, Prague, Czech Republic, May 2004.

M. Cuturi and J-P. Vert. Semigroup kernels on finite sets. In *Advances in Neural Information Processing*, Vancouver, Canada, December 2005.

S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

J. Eichhorn and O. Chapelle. Object categorization with SVM: kernels for local features. Technical report, MPI for Biological Cybernetics, 2004.

L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object cateories. In *Workshop on Generative Model Based Vision*, Washington, D.C., June 2004.

R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Madison, WI, 2003.

A. Frome, Y. Singer, and J. Malik. Image retrieval and classification using local distance functions. In B. Scholkopf, J.C. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

T. Gartner. A survey of kernels for structured data. *Multi Relational Data Mining*, 5(1):49–58, July 2003.

K. Grauman. *Matching Sets of Features for Efficient Retrieval and Recognition*. PhD thesis, MIT, 2006.

K. Grauman and T. Darrell. Fast contour matching using approximate Earth Mover's Distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Washington D.C., June 2004.

K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proceedings IEEE International Conference on Computer Vision*, Beijing, China, October 2005.

K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features. Technical Report MIT-CSAIL-TR-2006-020, MIT CSAIL, Cambridge, MA, March 2006.

K. Grauman and T. Darrell. Approximate correspondences in high dimensions. In B. Scholkopf, J.C. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems 19*, Cambridge, MA, 2007. MIT Press.

K. Grauman, G. Shakhnarovich, and T. Darrell. Inferring 3D structure with a statistical image-based shape model. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, October 2003.

E. Hadjidemetriou, M. Grossberg, and S. Nayar. Multiresolution histograms and their use for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):831–847, July 2004.

E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh. On the significance of real-world conditions for material classification. In *Proceedings of European Conference on Computer Vision*, Prague, Czech Republic, 2004.

A. Holub, M. Welling, and P. Perona. Combining generative models and fisher kernels for object recognition. In *Proceedings IEEE International Conference on Computer Vision*, Beijing, China, October 2005a.

A. Holub, M. Welling, and P. Perona. Exploiting unlabelled data for hybrid object classification. Technical report, NIPS 2005 Workshop on Inter-Class Transfer, Vancouver, Canada, December 2005b.

P. Indyk and N. Thaper. Fast image retrieval via embeddings. In *3rd International Workshop on Statistical and Computational Theories of Vision*, Nice, France, Oct 2003.

T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing*, Denver, CO, December 1999.

Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Washington, D.C., June 2004.

R. Kondor and T. Jebara. A kernel between sets of vectors. In *Proceedings of the International Conference on Machine Learning*, Washington, D.C., August 2003.

J. Lafferty and G. Lebanon. Information diffusion kernels. In *Advances in Neural Information Processing*, Vancouver, Canada, December 2002.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.

D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, January 2004.

S. Lyu. Mercer kernels for object recognition with local features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, San Diego, CA, June 2005.

J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of Optical Society of America*, 7(5):923–932, May 1990.

K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *Proceedings IEEE International Conference on Computer Vision*, Vancouver, Canada, July 2001.

P. Moreno, P. Ho, and N. Vasconcelos. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In *Advances in Neural Information Processing*, Vancouver, December 2003.

J. Mutch and D. Lowe. Multiclass object recognition using sparse, localized features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

F. Odone, A. Barla, and A. Verri. Building kernels from binary strings for image matching. *IEEE Transactions on Image Processing*, 14(2):169–180, February 2005.

M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

D. Roobaert and M. Van Hulle. View-based 3D object recognition with support vector machines. In *IEEE International Workshop on Neural Networks for Signal Processing*, Madison, WI, Aug 1999.

Y. Rubner, C. Tomasi, and L. Guibas. The Earth Mover's Distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, 1997.

T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426, 2007.

F. Shaffalitzky and A. Zisserman. Automated scene matching in movies. In *Proceedings, Challenge of Image and Video Retrieval*, London, U.K., July 2002.

G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, October 2003.

Y. Shan, S. Sawhney, and R. Kumar. Vehicle identification between non-overlapping cameras without direct feature matching. In *Proceedings IEEE International Conference on Computer Vision*, 2005.

A. Shashua and T. Hazan. Algebraic set kernels with application to inference over local image representations. In *Advances in Neural Information Processing*, Vancouver, Canada, Dec 2005.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. In *Proceedings IEEE International Conference on Computer Vision*, Beijing, China, October 2005.

J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, Oct 2003.

M. Swain and D. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

T. Tuytelaars and L. Van Gool. Content-based image retrieval based on local affinely invariant regions. In *3rd Intl Conference on Visual Information Systems*, Amsterdam, the Netherlands, June 1999.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

C. Wallraven, B. Caputo, and A. Graf. Recognition with local features: the kernel recipe. In *Proceedings IEEE International Conference on Computer Vision*, Nice, France, October 2003.

G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

J. Weston, B. Scholkopf, E. Eskin, C. Leslie, and W. Noble. Dealing with large diagonals in kernel matrices. In *Principles of Data Mining and Knowledge Discovery*, volume 243 of *SLNCS*, 2002.

J. Willamowski, D. Arregui, G. Csurka, C. R. Dance, and L. Fan. Categorizing nine visual classes using local appearance descriptors. In *ICPR Workshop on Learning for Adaptable Visual Systems*, 2004.

L. Wolf, S. Bileschi, and E. Meyers. Perception strategies in hierarchical vision systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

L. Wolf and A. Shashua. Learning over sets using kernel principal angles. *Journal of Machine Learning Research*, 4:913–931, Dec 2003.

H. Zhang, A. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, New York City, NY, June 2006.

H. Zhang and J. Malik. Learning a discriminative classifier using shape context distances. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Madison, WI, June 2003.