The Pyramid Match: Efficient Learning with Partial Correspondences

Kristen Grauman
Department of Computer Sciences
University of Texas at Austin
grauman@cs.utexas.edu

Abstract

It is often useful to represent a single example by a set of the local features that comprise it. However, this representation poses a challenge to many conventional learning techniques, since sets may vary in cardinality and the elements are unordered. To compare sets of features, researchers often resort to solving for the least-cost correspondences, but this is computationally expensive and becomes impractical for large set sizes. We have developed a general approximate matching technique called the *pyramid match* that measures partial match similarity in time *linear* in the number of feature vectors per set. The matching forms a Mercer kernel, making it valid for use in many existing kernel-based learning methods. We have demonstrated the approach for various learning tasks in vision and text processing, and find that it is accurate and significantly more efficient than previous approaches.

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 points. Likewise, in natural language processing, documents may be represented by bags of word meaning descriptors; 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, document, disease). The number of features per example varies, and within a single instance the component features may have no inherent ordering.

Learning with these sets (or bags) of features is challenging. Many conventional similarity measures and machine learning algorithms assume vector inputs, where each dimension corresponds to a particular global attribute for that instance, but general-purpose distances and kernels defined on $\Re^n$ inputs are not applicable in the space of vector sets. Existing approaches designed for 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 parametric distributions to the sets (which makes restrictive assumptions about the data and can also be expensive).

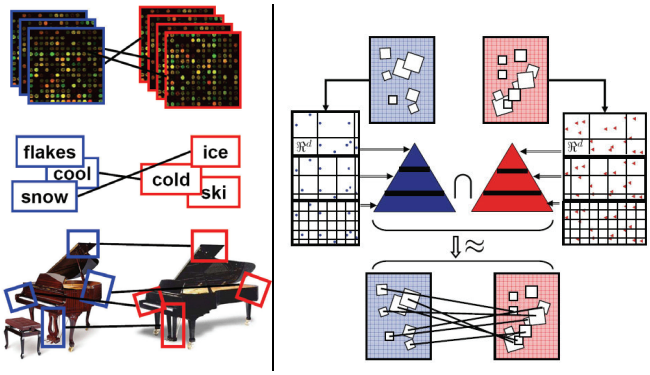


Figure 1: *Left:* A partial match between sets of features is useful to compare objects in various domains, including computational biology, language, and vision. *Right:* The pyramid match takes two sets of feature vectors as input (for instance, sets of local image patch descriptors as depicted here), maps the vectors to multi-resolution histograms, and intersects them to efficiently approximate the optimal partial matching (correspondence) between the original sets.

In recent work we developed the *pyramid match*—a new linear-time matching function over unordered feature sets—and showed how it allows set inputs to be used effectively and efficiently within the context of multiple learning problems (2005; 2007a; 2006; 2007b). The pyramid match approximates the similarity measured by the *optimal partial matching* between feature sets of variable cardinalities. Because the matching is partial, some features may be ignored without penalty to the overall set similarity. This tolerance makes the measure robust in situations where superfluous or "outlier" features may appear. The architecture of our method is quite simple: each feature set is mapped to a multi-resolution histogram (pyramid), and the pyramids are then compared using a weighted histogram intersection computation (see Figure 1).

We have shown that the pyramid match naturally forms a *Mercer kernel*, which means that it is appropriate to use with kernel-based learning methods that guarantee conver-

gence to a unique optimum only for positive-definite kernels (e.g., the Support Vector Machine). This connection to kernel methods is valuable, as it opens up a wealth of existing learning techniques for the set representation, including methods for discriminative classification, clustering, dimensionality reduction, and regression. We also provide approximation distortion bounds, which guarantee the pyramid match's expected accuracy relative to the optimal partial matching (2006b).

We have demonstrated our algorithm for a variety of tasks: supervised object recognition from sets of image patch features (2005), unsupervised discovery of visual categories (2006), 3-D human pose inference from sets of local contour features from monocular silhouettes, and documents' time of publication estimation from bags of local word features (2007b). In our results, approaches based on the pyramid match consistently show accuracy that is competitive with (or better than) the state-of-the-art while requiring dramatically less computation time. This complexity advantage frees us to consider much richer representations than were previously practical; for example, it removes the need to artificially limit the number of local descriptions used per image when learning visual categories.

In this paper we will overview the core algorithm, and then very briefly summarize some results for supervised learning of visual categories. Because our methods are generally applicable to efficient learning with unordered sets of features (from images or otherwise), we hope that this overview will prompt readers to consider places where the techniques might be useful in their own research.

Related Work

Space permits only a short review of previous work; please see our papers for more background and extensive discussions (2007b; 2006b).

Several researchers have designed kernel functions that can handle unordered sets as input (e.g., Kondor & Jebara 2003; Wallraven *et al.* 2003; Moreno *et al.* 2003). However, previous approaches suffer from prohibitive computational expense, make assumptions regarding the distribution of the features, are not positive-definite, and (or) are limited to sets of equal size.

Previous matching approximation methods have also considered a hierarchical decomposition of the feature space to reduce matching complexity (Charikar 2002; Indyk & Thaper 2003; Agarwal & Varadarajan 2004), and have in part inspired this work. However, they assumed equally-sized input sets, and could not compute partial matches. In addition, while previous techniques suffer from distortion factors that are linear in the feature dimension, we have shown how to alleviate this decline in accuracy by tuning the hierarchical decomposition according to the particular structure of the data (2007a). Finally, our approximation is unique in that it forms a valid Mercer kernel, and is useful in the context of various learning applications.

The Pyramid Match Algorithm

We consider a feature space $\mathcal{F}$ of d-dimensional vectors. The point sets we match will come from the input space S, which contains sets of feature vectors drawn from $\mathcal{F}$: $S = \{\mathbf{X} | \mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}\}$, where each feature $\mathbf{x}_i \in F \subseteq \Re^d$, and $m = |\mathbf{X}|$. Note that the point dimension d is fixed for all features in $\mathcal{F}$, but the value of m may vary across instances in S. The values of the vector elements have a maximal range D.

Given point sets $\mathbf{X}, \mathbf{Y} \in S$, with $|\mathbf{X}| \leq |\mathbf{Y}|$, the *optimal partial matching* π^* pairs each point in $\mathbf{X}$ to some unique point in $\mathbf{Y}$ such that the total distance between matched points is minimized: $\pi^* = \text{argmin}_\pi \sum_{\mathbf{x}_i \in \mathbf{X}} ||\mathbf{x}_i - \mathbf{y}_{\pi_i}||_1$, where π_i specifies which point $\mathbf{y}_{\pi_i}$ is matched to $\mathbf{x}_i$. For sets with m features, the Hungarian algorithm computes the optimal match in $O(m^3)$ time (Kuhn 1955).

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 a single partition encompasses the entire feature space. At some level along this gradation in bin sizes, any two particular points from two given point sets will begin to share a bin in the pyramid, 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—the size of the bin that two points share indicates the farthest distance they could be from one another. We show that a weighted intersection of two pyramids defines an implicit partial correspondence based on the smallest histogram cell where a matched pair of points first appears. The time to compute the pyramids as well as the weighted intersection is only linear in the number of features.

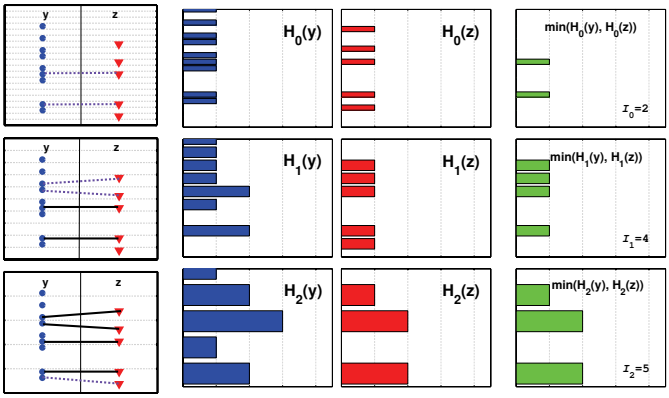
A histogram pyramid for input example $\mathbf{X} \in S$ is defined as: $\Psi(\mathbf{X}) = [H_0(\mathbf{X}), \ldots, H_{L-1}(\mathbf{X})]$, where $L = \lceil \log_2 D \rceil$, and $H_i(\mathbf{X})$ is a histogram vector formed over points in $\mathbf{X}$ using d-dimensional bins of side length 2^i. The bins in the finest-level histogram H_0 are small enough that each unique point in $\mathcal{F}$ falls into its own bin, and then the bin size increases until all points in $\mathcal{F}$ fall into a single bin at level $L-1$. Histograms are represented sparsely, meaning the full bin structure is never explicitly formed.[1]

The pyramid match $\mathcal{P}_\Delta$ similarity between two input sets $\mathbf{Y}$ and $\mathbf{Z}$ is defined as the weighted sum of the number of feature matches found at each level of their pyramids:

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

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 a pair of features that were not in correspondence at any finer resolution level.

[1]To enable accurate pyramid matching even with high-dimensional feature spaces, we have developed a variant where the bin structure is tailored to the distribution of the data (2007a).



(a) Point sets (b) Histogram pyramids (c) Intersections

Figure 2: An example pyramid match. Here, two 1-D feature sets are used to form two histogram pyramids. Each row corresponds to a pyramid level. In (a), set $\mathbf{Y}$ is on the left, and set $\mathbf{Z}$ is on the right; points are distributed along the vertical axis. Light 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; (c) shows their intersections. $\mathcal{P}_\Delta$ uses these intersection counts to measure how many new matches occurred at each level. Here, $\mathcal{I}_i = \mathcal{I}(H_i(\mathbf{Y}), H_i(\mathbf{Z})) = 2, 4, 5$ across levels, so the number of new matches counted are $N_i = 2, 2, 1$. ($\mathcal{I}_{-1} = 0$ by definition.) The sum over N_i, weighted by $w_i = 1, \frac{1}{2}, \frac{1}{4}$, gives the pyramid match similarity.

The 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. 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 2, there are two points matched at the finest scale, two new matches at the medium scale, and one at the coarsest scale.

To calculate N_i, we use histogram intersection, which measures the "overlap" between two histograms' bin counts: $\mathcal{I}(\mathbf{A}, \mathbf{B}) = \sum_{j=1}^{r} \min(\mathbf{A}^{(j)}, \mathbf{B}^{(j)})$, where $\mathbf{A}$ and $\mathbf{B}$ are histograms with r bins, and $\mathbf{A}^{(j)}$ denotes the count of the j^{th} bin. The intersection value effectively counts the number of points in two sets that match at a given quantization level, i.e., 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 levels' intersections:

$$N_i = \mathcal{I}(H_i(\mathbf{Y}), H_i(\mathbf{Z})) - \mathcal{I}(H_{i-1}(\mathbf{Y}), H_{i-1}(\mathbf{Z})), \quad (2)$$

where H_i refers to the i^{th} component histogram generated by Ψ. The measure does not explicitly search for similar points, and 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.

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.[2] This reflects a geometric bound on the maximal distance between any two points that share a particular bin. 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.

From Eqns. 1 and 2, we define the (un-normalized) pyramid match:

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

We normalize this value by the product of each input's self-similarity to avoid favoring larger input sets. In order to alleviate quantization effects from the discrete histogram bins, we combine the values resulting from multiple matches formed under pyramids with bins shifted by amounts chosen uniformly at random from $[0, D]$.

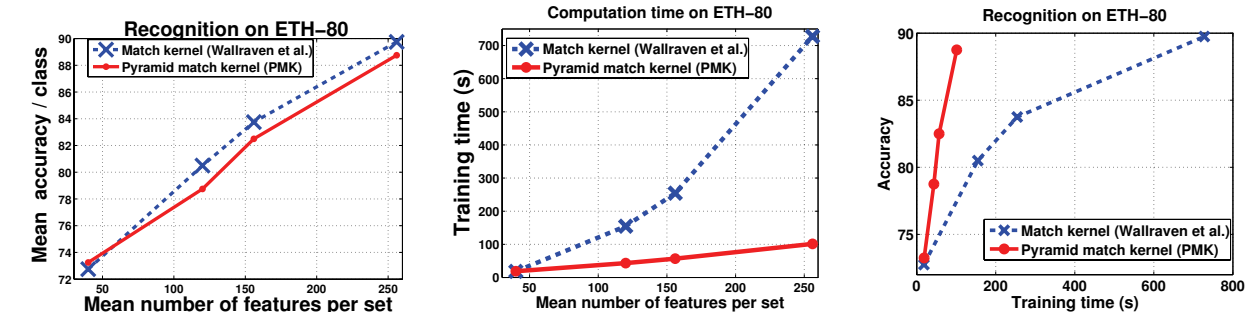
In fact, the random shifts and the bin weights defined above are not solely based on intuition. Both components allow us to show that the approximation error for the pyramid match cost with uniformly shaped bins is bounded in the expectation by a factor of $C \cdot d \log D + d$ (2006b). We have also proven that the pyramid match naturally forms a Mercer kernel, meaning that it corresponds to a dot product in some feature space and can be used as the basis for any kernel method (2005). The pyramid match remains a Mercer kernel for any choice of weights in which $w_i \geq w_{i+1}$ (2006b).

The optimal partial matching requires $O(m^3)$ time for sets with $O(m)$ features, which severely limits the practicality of large input sizes. Even a suboptimal greedy matching requires $O(m^2 \log m)$ time, since all pairwise distances between points in the two sets must be computed and sorted. In contrast, our approach requires only $O(m \log D)$ time to compute both the pyramids as well as a matching, for pyramids with $L = \log D$ levels. In practice, this translates to speedups of several orders of magnitude relative to the optimal match for sets with $m \approx 1000$ and $L \approx 10$ (2006b).

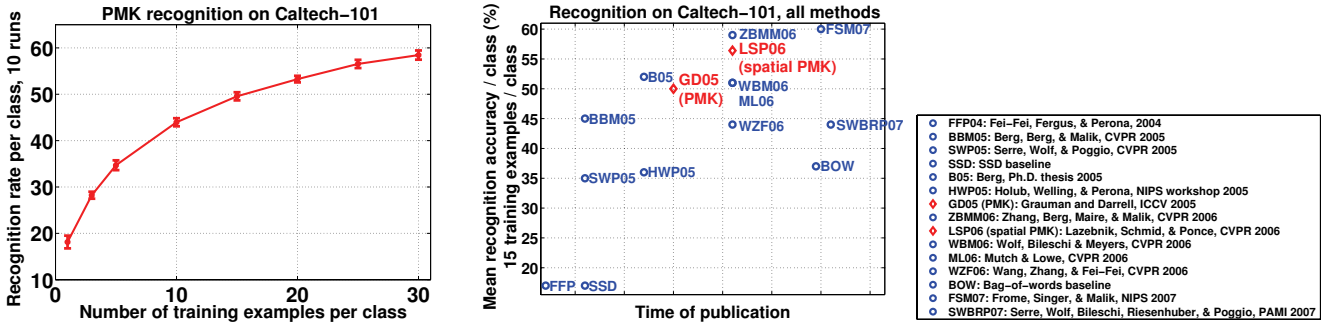
Learning Visual Categories

Object recognition is a challenging problem due to the broad variety in illumination, viewpoint, occlusions, clutter, and intra-class appearance that images of the same object class will exhibit. Much recent work shows that decomposing an image into its component parts (or *local features*) grants resilience to common transformations and variations. The idea is that strong similarity between multiple local portions of related images may often be discovered even when globally the images appear quite different. Typically, an *interest operator* is used to identify numerous salient regions in an image; then, for each region, a (vector) feature descriptor is formed. Possible salient points include pixels marking high contrast (edges), or points selected for a region's repeatability at multiple scales. Descriptors may be lists of pixel values within a patch, or histograms of oriented contrast within the regions, for example. Whatever the local representation

[2] To instead use the matching as a cost function, $w_i = d2^i$.



(a) ETH-80 data set



(b) Caltech-101 data set

Figure 3: Object recognition with the PMK and other methods on the ETH-80 (top row) and Caltech-101 (bottom row) data sets.

choice, the result is one large set of local descriptor vectors per image, often numbering on the order of $m = 2000$.

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. Given a collection of images with feature sets $\mathbf{X}_1, \ldots, \mathbf{X}_N$, the $N \times N$ *kernel matrix* $\mathbf{K}$ specifies all pairwise partial match similarities, with $\mathbf{K}_{ij} = \mathcal{P}_\Delta(\mathbf{X}_i, \mathbf{X}_j)$. This matrix specifies an embedding for the original images that captures their relations according to the strength of their partially matching features; we have shown how to use this embedding to learn visual categories from labeled (2005) or unlabeled (2006) images. Here we overview some results using labeled images.

Given a collection of images labeled according to which object categories they contain, we can use the pyramid match to build a Support Vector Machine (SVM) (Vapnik 1998). An SVM is a discriminative classifier that uses the mutual positions of the labeled training examples to identify the hyperplane that separates examples (in some feature space) such that the margin between different categories is maximal. The optimization is cast as a quadratic programming problem involving the kernel matrix; we are guaranteed to find a unique optimal solution using the PMK because it is a Mercer kernel.

In experiments with the publicly available ETH-80 and Caltech-101 databases, we have found that pyramid match category learning offers very strong accuracy at a significantly lower computational cost than other state-of-the-art approaches. For our image experiments, we decompose images into local patches and describe each patch with the local invariant descriptor called SIFT (Lowe 2004). More details are in a recent journal paper (2007b).

Note that for both the Caltech-101 and ETH-80 databases, there is a single object of interest in each image, and it appears prominently relative to the background. In this case the PMK matching is computed between all features in the two images that are being compared. For cases where novel images may contain multiple objects of interest or widely varying amounts of clutter, we expect it would be suitable to compute the PMK matching within multi-scale image windows; this is to be verified experimentally in future work.

A performance evaluation given by Eichhorn & Chapelle compares the set kernels of Kondor & Jebara, Wolf & Shashua, and Wallraven *et al.* using SVMs and images from the ETH-80 database of eight object classes. Tested under the same conditions, the PMK performs comparably to the others at their best for this data set, but is much more efficient. In fact, the ability of a kernel to handle large numbers of features can be critical to its success. Figure 3 (a) compares the $O(m)$-time PMK with the $O(m^2)$-time match kernel of Wallraven *et al.*, for increasingly larger feature set sizes obtained by decreasing the saliency threshold of the

interest operator. For both methods, recognition accuracy benefits from having more features per image with which to judge similarity (left plot), but computing a kernel matrix for the same data is significantly faster with the PMK (middle plot). Allowing the same amount of training time for both methods, the PMK produces much better recognition results (right plot).

The well-known Caltech-101 database contains 101 diverse object categories, and is currently the largest benchmark data set available; it is challenging due to the large number of categories as well as the significant amount of intra-class appearance variation it contains. The lefthand plot in Figure 3 (b) shows the multi-class category recognition results using the PMK, for varying numbers of training examples. Note that chance performance would be just 1%. Experiments comparing the PMK recognition accuracy to an optimal partial matching kernel have shown that negligible loss in accuracy is traded for speedup factors of several orders of magnitude (2006b).

The righthand plot in Figure 3 (b) shows all results on this data set as a function of time since it was released, including those published more recently by other authors. From this comparison we can see that even with its extreme computational efficiency (a matching requires just 0.0001 seconds), the PMK achieves results that are very competitive with the state-of-the-art. We obtain 50% accuracy on average ($\sigma = 0.9\%$ over 10 runs) when using the standard 15 training examples per category. In addition, Lazebnik *et al.* (2006) have shown that using the PMK with sets of spatial features also yields very good accuracy, 56.4%. These results are among the very best reported to-date on the data set—only a few percentage points away from the most accurate result of 60%, which was obtained recently by Frome *et al.* (2007) using discriminative distance functions that compute matches between local geometric blur features.

In addition, our approach's extreme efficiency gives it a clear practical advantage. Classifying a novel example from this data set with the PMK requires just a fraction of a second, whereas methods that compute explicit correspondences (Frome, Singer, & Malik 2007; Zhang *et al.* 2006) require about one minute; in the time that these methods recognize a single object, the PMK recognizes several hundred objects.

Conclusions

We have developed a new linear-time partial matching function that handles unordered, variably-sized sets of features; in recent work we have shown its suitability for various learning problems, including supervised learning of visual categories as discussed here. In ongoing work we are developing an indexing method that performs partial-match similarity search in sub-linear time, and exploring ways in which discriminative correspondence distances may be directly learned.

We hope that this opportunity to present our work to the AAAI community will allow us to connect with researchers outside of computer vision who face similar matching and learning problems. To facilitate the use of our methods, we have made source code available at `http://people.csail.mit.edu/jjl/libpmk/`.

References

Agarwal, P., and Varadarajan, K. 2004. A Near-Linear Algorithm for Euclidean Bipartite Matching. In *Symposium on Computational Geometry*.

Charikar, M. 2002. Similarity Estimation Techniques from Rounding Algorithms. In *Symposium on Theory of Computing*.

Eichhorn, J., and Chapelle, O. 2004. Object Categorization with SVM: Kernels for Local Features. Technical report, MPI.

Frome, A.; Singer, Y.; and Malik, J. 2007. Image Retrieval and Classification Using Local Distance Functions. In *Advances in Neural Information Processing Systems (NIPS) 19*.

Grauman, K., and Darrell, T. 2005. The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features. In *Proceedings of the IEEE International Conference on Computer Vision*.

Grauman, K., and Darrell, T. 2006. Unsupervised Learning of Categories from Sets of Partially Matching Image Features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Grauman, K., and Darrell, T. 2007a. Approximate Correspondences in High Dimensions. In *Advances in Neural Information Processing Systems (NIPS) 19*.

Grauman, K., and Darrell, T. 2007b. The Pyramid Match Kernel: Efficient Learning with Sets of Features. *Journal of Machine Learning Research (to appear)*.

Grauman, K. 2006b. *Matching Sets of Features for Efficient Retrieval and Recognition*. Ph.D. Dissertation, MIT.

Indyk, P., and Thaper, N. 2003. Fast Image Retrieval via Embeddings. In *International Workshop on Statistical and Computational Theories of Vision*.

Kondor, R., and Jebara, T. 2003. A Kernel Between Sets of Vectors. In *Proceedings of International Conference on Machine Learning*.

Kuhn, H. 1955. The Hungarian Method for the Assignment Problem. *Naval Research Logistic Quarterly* 2:83–97.

Lazebnik, S.; Schmid, C.; and Ponce, J. 2006. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Scene Categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Lowe, D. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60(2):91–110.

Moreno, P.; Ho, P.; and Vasconcelos, N. 2003. A Kullback-Leibler Divergence Based Kernel for SVM Classification in Multimedia Applications. In *Advances in Neural Information Processing (NIPS) 16*.

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

Wallraven, C.; Caputo, B.; and Graf, A. 2003. Recognition with Local Features: the Kernel Recipe. In *Proceedings of the IEEE International Conference on Computer Vision*.

Wolf, L., and Shashua, A. 2003. Learning Over Sets Using Kernel Principal Angles. *Journal of Machine Learning Research* 4:913–931.

Zhang, H.; Berg, A.; Maire, M.; and Malik, J. 2006. SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.