Local Bi-gram Model for Object Recognition

MSR-TR-2007-54

Xiangyang Lan
Cornell University
xylan@cs.cornell.edu

C. Lawrence Zitnick
Microsoft Research
larryz@microsoft.com

Richard Szeliski
Microsoft Research
szeliski@microsoft.com

Abstract

In this paper, we describe a model-based approach to object recognition. Spatial relationships between matching primitives are modeled using a purely local bi-gram representation consisting of transition probabilities between neighboring primitives. For matching primitives, sets of one, two or three features are used. The addition of doublets and triplets provides a highly discriminative matching primitive and a reference frame that is invariant to similarity or affine transformations. The recognition of new objects is accomplished by finding trees of matching primitives in an image that obey the model learned for a specific object class. We propose a greedy approach based on best-first-search expansion for creating trees.

Experimental results are presented to demonstrate the ability of our method to recognize objects undergoing non-rigid transformations for both object instance and category recognition. Furthermore, we show results for both unsupervised and semi-supervised learning.

1. Introduction

Matching a discrete set of local patch-based features is a useful technique for object recognition. The effectiveness of these methods relies mainly on the discriminative power of the features' descriptors. This is best demonstrated by the effectiveness of "bag of words" approaches [15, 17]. Recently, methods that additionally model the spatial relationship of features have shown improved results. This is especially true for category recognition where the appearance of features across intra-class objects can vary more dramatically than object instance recognition. Several types of spatial models have been developed, including the constellation model [8, 19], star models [5, 6], rigid 3D models [16], and image-centric or warping techniques [3, 7]. These methods all create a global model for an object, whether they are parts-based, image-based or full 3D models.

In this paper, we propose using a local spatial model without any explicit global model of the object. We only model the relationship of neighboring features without knowledge of their global context. An illustrative analogy between our approach and the task of language modelling can be made. In language modeling, the goal is to model the spatial relationship of words in sentences. Most global methods attempt to model the entire sentence structure, noun phrase, verb phrase, etc. Local models only attempt to model the structure of neighboring words, such as the n-gram model [2]. An n-gram model uses the previous $n-1$ words in a sentence to predict the probability of the next word. In the language modeling community, these simplistic local techniques have shown respectable results in comparison to the more complex global techniques. Similarly to the n-gram language model, we propose a bi-gram model for object recognition. That is, each "word" or matching primitive in our model is only dependent on a single word preceding it in the model.

A critical design element is the choice of matching primitive. This primitive should be both discriminative between different types of objects and predictive of the local structure within an object. Previous methods [3, 5, 8] typically use single features, such as SIFT [12] for this purpose. In addition to using single features, we propose using sets of two or three neighboring features as our matching primitive [17, 11]. By combining features into sets, we increase their discriminative power over single features. Multiple feature positions can also be used to compute robust similarity and affine transformations to better predict the local position of neighboring features.

Unlike the 1D problem of language modeling, in which a strict ordering of words is enforced in a string, words or matching primitives in object recognition have no explicit order. Moreover, multiple matching primitives can be dependent on each other. To handle this more complex relationship, we propose modeling the structure of objects using a tree instead of a string. This provides an efficient representation with an inherent ordering that allows for multiple

Figure 1. Example of two sets of feature triplets assigned to the same words from the Caltech 101 face data set. Upper row corresponds to one word, and lower row corresponds to another.

matching primitives to be dependent on a single parent.

Our algorithm is split into several steps. First during the learning phase, a model of each object is found from a set of training images. This model consists of a set of frequently occurring match primitives along with the prediction of their neighboring primitives. Only the local relationships are stored, without the global context from which they came. Second given a testing image, a set of matching primitives are found. Finally, we search for a tree structure of matching primitives that obey the local model found in the learning phrase.

An outline of the paper is as follows. We first describe how we create an object model given a set of training images in Section 2. Next, in Section 3 we discuss how we use these models to recognize objects in images. This includes two steps, finding a coherent tree of matching primitives and predicting the object given the tree structure. Finally, we show results on several object categories.

2. Object Model

In this section, we describe our local bi-gram object model. This model only represents the relationships between neighboring match primitives in an image. These local relationships are based on the features' descriptors, as well as their local positions and orientations. By only modeling the local interactions of features, a compact model for deformable objects with large variation of appearance can be created.

We describe our object model in two steps. First, we describe our matching primitives and the vocabulary of words created from them. Second, we learn the local spatial relationships between the words.

2.1. Feature Singlets, Doublets and Triplets

A matching primitive should be both discriminative between different types of objects, and repeatable within a specific class of object. Previous research has used single features such as SIFT [12], corner detectors [9, 14], maximally stable extremal regions [13, 15], and salient regions [10]. The use of single features provides a high-level of repeatability. However, the occurrence of these features is not unique to a specific object. This is especially true when the features are discretized [15, 18] into a fixed number of clusters. To increase the discriminative power of matching primitives, multiple features can be used, such as doublets [17] or triplets [11]. While this does reduce their repeatability, their uniqueness to a single object class is increased.

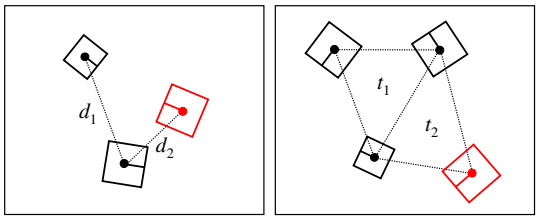
We propose three approaches using either sets of one, two or three SIFT [12] features called singlets, doublets and triplets as our matching primitive. The use of single SIFT features has shown good performance for a variety of object recognition tasks [12, 17, 18]. The grouping of SIFT features provides additional contextual information. By analogy, in language models, seeing "white" or "house" doesn't necessarily identify an article's topic as politics, but seeing "white" and "house" closely together greatly increases this probability. Similarly in object recognition, the co-occurrence of features increases their discriminative power over each individual feature. Doublets and triplets are found by grouping features that lie within a certain distance of each other both spatially and in scale space. More specifically, if a feature f_i has a scale s_i, the other features in the set must lie within distance αs_i of each other in image space and $\pm\beta$ in log scale space.

To increase matching efficiency and to reduce model complexity, we discretize the set of feature descriptors into a set of fixed size [15, 17, 18]. This set is created by clustering the training feature descriptors using K-means. For our experiments we set $K = 1000$. As reported in [17], this approximate size provides a good tradeoff between repeatability and descriptiveness.

Our vocabulary is created from the n-tuple ($n \in \{1, 2, 3\}$) of cluster indices assigned to the feature descriptors. That is, for doublet d_i, with two features f_i^1 and f_i^2 assigned to clusters c_i^1 and c_i^2, we assign word $w_i = \{c_i^1, c_i^2\}$, $w_i \in C \times C$. We assume indices of the clusters are in numerical order, i.e. $c_i^1 < c_i^2$. To remove matching ambiguity, all doublets with repeated cluster indices are removed. Singlet and triplet words are similar except they contain 1 or 3 distinctive cluster indices. Using this technique we can group singlets, doublets and triplets assigned to the same words together. Examples of triplets assigned to the same words are shown in Figure 1. The largest theoretical size of our vocabulary is $\binom{1000}{2}$ for doublets and $\binom{1000}{3}$ for triplets. In practice, an object will only contain a small fraction of these sets.

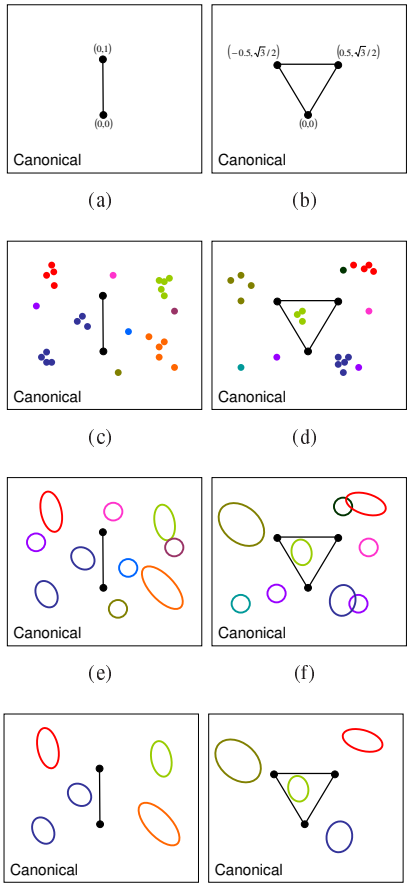
2.2. Learning Local Relationships

Our next task is learning the local relationships of neighboring matching primitives. Two doublets are said to be neighboring if they share a single feature (Figure 2(a)) while triplets share two features (Figure 2(b)). Singlets are neighboring if they lie within a certain distance in image and scale



(a) (b)

Figure 2. Example of neighboring doublets (a) and triplets (b). The transition features from doublet d_1 to d_2 and triplet t_1 to t_2 are shown in red.



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 3. Illustration of doublet (a) and triplet (b) positions in the canonical frame, projection of transition features from neighbors, different colors representing transition features from different K-mean cluster (c, d), clustering after mean shift (e, f), and the final model after thresholding cluster size (g, h).

space (same criterion as the construction of doublets). In each case, there exists one feature in each set not shared by the other set. This feature is called the transition feature of its set with respect to the transition from the other set. More specifically, the transition feature of doublet d_i with respect to the transition from doublet d_j is denoted as $f_i(j) \in \{f_i^1, f_i^2\}$, and similarly for pairs of singlets and triplets. Example transition features can be seen in red in Figure 2.

Our local bi-gram model consists of a set of transition probabilities between neighboring primitives, for a given object or an object category O_l. For simplicity, we first discuss how to compute the doublet transition probability. Since neighboring doublets share all but one feature between them, the task of computing the probability of $p(d_i|d_j, O_l)$ can be reduced to computing $p(f_i(j)|d_j, O_l)$ for all $d_i \in N_j$, where N_j are all neighbors of d_j. A feature f contains its location x, orientation θ and cluster index c, $f = \{x, \theta, c\}$. So a doublet d_j consists of two features f_j^1 and f_j^2 with two positions x_j^1 and x_j^2, orientations θ_j^1 and θ_j^2, and cluster indices c_j^1 and c_j^2. In our model the doublet d_j is identified by its corresponding word $w_j = \{c_j^1, c_j^2\}$. Thus,

$$p(f_i(j)|d_j, O_l) = \\ p(\{x_i(j), \theta_i(j), c_i(j)\}|\{x_j^1, x_j^2, \theta_j^1, \theta_j^2, w_j\}, O_l) \quad (1)$$

Spatial relationships are modeled conditional on word assignment, and are enforced in a canonical frame, where x_j^1 projects to (0,0) and x_j^2 to (0,1) (Figure 3(a)). This provides invariance up to a similarity transformation (translation, rotation and scale). Let, $\mathbf{A}_j$ be the 3×3 matrix transforming homogeneous coordinates x_j^1 and x_j^2 to (0,0) and (0,1) respectively. If $\acute{x}_i(j)$ and $\acute{\theta}_i(j)$ are feature $f_i(j)$'s position and orientation in the canonical frame defined by $\mathbf{A}_j$,

$$p(f_i(j)|d_j, O_l) = p(\{\acute{x}_i(j), \acute{\theta}_i(j), c_i(j)\}|w_j, O_l) \quad (2)$$

We learn the value of equation (2) using the training data set. First, a set of transition features $F(w_j)$ is found from all neighboring doublets to doublets which are assigned to the word w_j in the training data. Next, each transition feature $f_k \in F(w_j)$ is projected into the canonical frame based on the neighboring doublets' relative positions (Figure 3(c)).

To represent the set of possible transition features, several models may be used, such as a local Gaussian perturbation model or a Gaussian Mixture Model (GMM). We propose using a GMM. A GMM allows us to model groups of transition features assigned to the same words using a Gaussian with computed means and variances.

Since the number of Gaussians needed to represent the distribution is unknown and varies across doublets, we cluster the transition features using mean shift [4]. For mean shift, we cluster features based on their appearance, position and orientation. Our iterative update equation for mean shift is,

$$\{\acute{x}_k^t, \acute{\theta}_k^t\} = \frac{\Sigma_i \acute{x}_i \mathcal{N}(\acute{x}_i; \acute{x}_k^{t-1}, \sigma_x^2)\mathcal{N}(\acute{\theta}_i; \acute{\theta}_k^{t-1}, \sigma_\theta^2)\delta(c_i, c_k)}{\Sigma_i \mathcal{N}(\acute{x}_i; \acute{x}_k^{t-1}, \sigma_x^2)\mathcal{N}(\acute{\theta}_i; \acute{\theta}_k^{t-1}, \sigma_\theta^2)\delta(c_i, c_k)} \quad (3)$$

The function $\mathcal{N}$ is the standard Normal distribution over the spatial coordinates in the canonical frame. Standard deviations of $\sigma_x = 0.4$ and $\sigma_\theta = 0.25\pi$ are used for position and orientation in the canonical frame. Features are clustered together based on their assigned cluster indices, i.e. $\delta(c_i, c_k) = 1$ if $c_i = c_k$ and 0 otherwise. Thus, only features assigned to the same cluster indices with similar positions and orientations are clustered together. An illustrative example of this clustering can be seen in Figure 3(e).

To ensure the compactness of the model, only doublets which occur at least ψ times in the training data are kept. Further, clusters of transition features that have less than ω members are discarded (Figure 3(g)). In our experiments $\psi = 3$ and $\omega = 3$.

After mean shift, the transition probabilities are represented by a set of Gaussians, each of which is factored into appearance, position and orientation components. This results in our transition probability for $p(f_i(j)|d_j)$ being equal to the summation over all the Gaussian components index by a,

$$p(f_i(j)|d_j, O_l) = \Sigma_a p(a)\Delta(f_i(j), c_a) \mathcal{N}(\acute{x}_i(j); \acute{x}_a, \Sigma_x(a))\mathcal{N}(\acute{\theta}_i(j); \acute{\theta}_a, \sigma_\theta(a)^2) \quad (4)$$

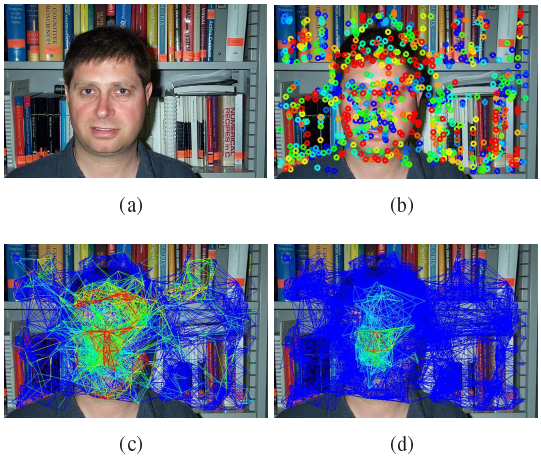
The similarity in appearance is computed using $\Delta(f_i(j), c_a)$, which is described later in Section 4. The spatial covariance matrix $\Sigma_x(a)$ and orientation variance $\sigma_\theta(a)^2$ are computed separately for each Gaussian component in the GMM based on the training data. $p(a)$ is the prior probability for each component in the GMM.

The transition probabilities for neighboring feature singlets and triplets are learned in the same manner as those learned for doublets with the only exception being how the transform $\mathbf{A}_j$ is computed. The similarity transform for the canonical frame of singlets is computed from the feature's position, scale and orientation. For triplets, the three feature positions are projected to the positions $(0, 0), (-0.5, \sqrt{3}/2), (0.5, \sqrt{3}/2)$ corresponding to an equilateral triangle in the canonical frame (Figure 3(b)). This provides invariance to an affine transformation, and not just a similarity transformation. Once the transition features are projected into the canonical frame, the GMM is computed as described above (Figure 3(d,f,h)).

Figure 4, illustrates the set of singlets, doublets and triplets found using our model on the Caltech 101 face data set. The color encodes the frequency of occurrence for each matching primitive in our model. Notice the triplets with high occurrence are centered on the face and absent from the background image. Singlets are less discriminative and scattered over the face and background image. Using 216 training images with approximately 1000 features each, the size of the models found is summarized in Table 1.

Model	# of MPs	# of Trans. Features
Singlet	573	35.4
Doublet	12261	16.1
Triplet	7426	9.2

Table 1. Size of models for Caltech 101 face data set. The table shows the number of matching primitives with unique words and the average number of transition features per matching primitive.



(a) (b) (c) (d)

Figure 4. Frequency of occurrence for singlets (b), doublets (c) and triplets (d) in the face model generated from the Caltech 101 face data set. Blue = low occurrence, Red = high occurrence. Singlets, doublets and triplets are represented by circles, lines and triangles respectively.

3. Finding Objects

Our goal is to find objects in a new image given a set of matching primitives. Given our local object model, this task is transformed into finding a set of neighboring primitives that obey our model. Finding this set of neighboring matching primitives could be accomplished with several techniques. For instance, a Markov Random Field with edges between neighboring primitives could be created. Loopy belief propagation could then be used to update the likelihoods of each primitive belonging to an object. In this paper, we use a more efficient and simpler technique using trees of matching primitives. The children in a tree allow us to model the many possible neighbors a primitive might have in its 2D neighborhood. As a result, our goal is transformed into finding a tree of matching primitives with high likelihood of belonging to a specific object.

This task is split into two steps. First, finding a tree of neighboring matching primitives (singlets, doublets or triplets) that have a high probability given our object models. Second, given a set of matching primitives, determining the object to which they belong. We begin by describing how to compute the probability of an object given a tree, followed by a greedy algorithm for finding trees.

3.1. Probability of an Object Given a Tree

Let us assume we have a tree G of matching primitives. For clarity, we'll assume our matching primitive is a doublet for this section and the next. However, the same techniques also apply to singlets and triplets. Using Bayes Theorem we find,

$$p(O_l|G) \propto p(G|O_l)p(O_l) \tag{5}$$

The prior probabilities $p(O_l)$ for the objects O_l can be assumed to be uniform, unless addition information is known. For instance, if it is known the image was taken indoors, objects that are more likely to be found indoors can be given higher prior probabilities. Besides the objects that have been modeled, there is also a prior background probability. This background probability $p(BG)$ typically has a value much higher than the objects' prior probabilities, since for most images the majority of doublets and triplets will lie on unknown objects.

In the tree, each doublet d_i has a parent $P(d_i)$, with the root node of the tree having no parent $P(d_i) = \emptyset$. Given its tree structure, the likelihood of G can be computed as a product of conditional probabilities,

$$p(G|O_l) = \prod_i p(d_i|P(d_i), O_l) \tag{6}$$

We refer to the probability of the root node of the tree as the initial probability, and all remaining probabilities as transition probabilities within the tree.

Since the tree provides an implicit ordering of the primitives, the above model can be viewed as a bi-gram model applied to a tree structure. That is, each primitive depends on its preceding primitive, i.e. its parent in the tree.

The transition probabilities can be computed using equation (4) from our object model. The initial probability $p(d_i|O_l)$ of the root node is the normalized frequency count of doublets assigned to the same word as d_i within the object training set.

3.2. Finding Trees of Matching Primitives

Previously, we described how to compute the likelihood of an object given a tree. In this section, we describe a method for finding a tree with matching primitives belonging to a single object.

We construct a tree using a greedy algorithm. We begin by computing the initial probability for each matching primitive present in the test image. The initial probabilities are computed by marginalizing over all possible objects O in our database:

$$p(d_i|I) = \sum_l p(d_i|O_l)p(O_l|I) \tag{7}$$

We use the doublet with the highest likelihood as the root node in our tree. After the tree has grown to its full size, the next most likely doublet is found that wasn't used in a previous tree. This process is repeated until there are no remaining doublets.

After the root node is found, we iteratively expand our tree, picking the most likely doublet to add to our tree at each time step. Given our current tree G_t, we want to compute the most likely doublet d_j given the current likelihood of the objects O. This can be computed by marginalizing over all objects,

$$p(d_j|G_t) = \sum_l p(d_j|P(d_j), O_l)p(O_l|G_t) \tag{8}$$

Our new tree G_{t+1} is created by adding the doublet d_j to the tree G_t that maximizes equation (8). Once a new doublet has been added to a tree, the likelihood of the objects within O can be updated using equations (5) and (6) .

Once the likelihoods of the objects are updated, the likelihoods of all possible neighboring doublets to the tree are also updated. New doublets are then iteratively added to the tree until no neighboring doublets in the image have a higher likelihood of belonging to an object than to the background. Since the number of possible doublets can be quite large, empirically computing the background probability is infeasible. In practice, setting the background probability to a constant ϵ provides reasonable results.

If the final probability of a tree belonging to an object is greater than it belonging to the background, an object is said to be found. Since the prior probability of the background is typically much higher than that of other objects, a tree must grow to a moderate size before an object's probability is higher than the background.

4. Implementation Details

In this section, we discuss the implementation details of our algorithm. For interest point detectors, we used either SIFT [12] or Harris corner detection across multiple scales [1]. Experimentally, we found the feature's orientation computed using these methods to be unstable for object category recognition. In our experiments, the orientation was computed by finding the angle at which the feature's descriptor best matched that a descriptor cluster mean. For K-means clustering of descriptors, the assignment to clusters was also found by matching over all orientations. For efficiency, only 16 orientations were used.

The function $\Delta(f_i, c)$ in equation (4) is computed by finding the three closest cluster centers $c_i(1)$,$c_i(2)$ and $c_i(3)$ to f_i's descriptor over all orientations. If $d(f_i, c)$ is the distance between f_i's descriptor and cth cluster mean,

$$\Delta(f_i, c) = 1 - \max(0, \frac{d(f_i, c) - d(f_i, c_i(1))}{d(f_i, c_i(3)) - d(f_i, c_i(1))}) \quad (9)$$

Thus, the closest cluster $c_i(1)$ has a value of 1 while the second cluster $c_i(2)$ varies between 0 and 1. For all other values of c, $\Delta(f_i, c) = 0$. Using this soft assignment technique, we can reduce the occurrence of misclassifying feature descriptors to cluster indices.

For our experiments, the probability of a transition feature belonging to the background, used to decide when a tree should stop growing, was set to $p(f_i(j)|BG) = 0.05$ for singlets and $p(f_i(j)|BG) = 0.2$ for doublets and triplets. The value is lower for singlets since they occur in general with higher frequency.

5. Experimental Results

To demonstrate the matching and learning performance of our algorithm we provide experiments on several objects using both semi-supervised and unsupervised learning.

5.1. Semi-supervised Learning

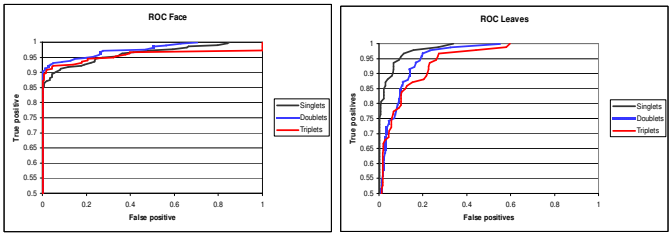
Our first experiment tests feature singlets, doublets and triplets on on two data sets: Faces 1999 and Leaves 1999 (www.vision.caltech.edu/html-files/archive.html). These data sets were chosen because interest points could be reliably found on them. The SIFT interest point detector was used for the face data set. For the leaves data set, the Harris corner detector was used since it produces more stable interest points at object boundaries. We use the same experimental setup as [8] and [3] for comparison. The face data set consists of 216 training images and 217 testing images. The leaves data set was randomly split into two groups of 93 images. For generating ROC curves and Equal Error Rate (EER) values, 300 images from the Caltech background data set were used. Our results are summarized in Tables 2 and 3. In all but one test, large trees where found with the number of unique features in the trees ranging from 55.9 to 31.7. In contrast, tree sizes for background images range from 0 to 4 features. For the leaf data set, few neighboring triplets were found and tree sizes remained small. The EER are similar for all matching primitives with doublets and singlets slightly outperforming the others on the face and leaves data sets respectively. For the face data set, images in which the lighting was similar (i.e. a flash was used) produced repeatable interest points and large matching primitive trees were found. Matching primitives associated with faces under different illuminations had fewer training images and were less likely to be added to the model. This resulted in reduced recognition results. For comparison, the following EER rates were achieved using other methods: 96.4% [8], 98.2% [5] and 98.2% [3]. A standard bag-of-words technique using normalized frequency counts of words achieved EERs of 64.5%, 70.0% and 96.8% for singlets, doublets and triplets respectively. The high result for triplets demonstrates their discriminative power even without the use of trees. For the leaf data set an EER of 92.1%, similar to our singlet result, was achieved by [3], while bag-of-words produced EERs ranging from 71.0 to 81.7. ROC curves for our results can be seen in Figure 5. Example face and leaf detections for singlets, doublets and triplets are shown in Figures 6 and 7.

MP	EER	BoW ERR	Tree size #	BG tree size
Singlet	91.2	64.5	55.9	4.2
Doublet	93.1	70.0	55.9	2.1
Triplet	92.2	96.8	39.7	0.2

Table 2. Results for Faces 1999 data set: EER, EER for bag-of-words model, the average number of unique features in trees for face images, and the average number of unique features in trees for background images.

MP	EER	BoW EER	Tree Size	BG Tree Size
Singlet	92.5	71.0	43.8	4.3
Doublet	87.1	80.7	31.7	4.2
Triplet	86.0	81.7	5.6	0.1

Table 3. Results for Leaves 1999 data set: EER, EER for bag-of-words model, the average number of unique features in trees for face images, and the average number of unique features in trees for background images.

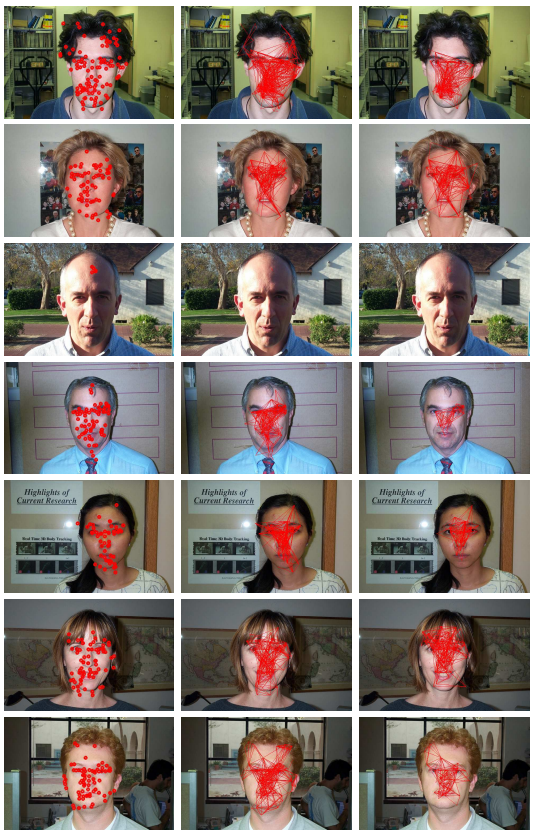


(a) (b)

Figure 5. ROC curves for Face 1999 and Leaves 1999 data sets for singlets, doublets and triplets.

5.2. Unsupervised Learning

To test the ability of our algorithm to handle noisy data, we reduced the face training data set to 100 images. Additionally, we added 150 background images to the training data set to act as distracters. Our results are shown in Table 4 and Figure 8. For singlets the results are almost identical, even with only 40% of the training data containing faces.



(a)

Figure 6. Example results for Face 1999 data set. Largest tree found is shown in red for singlets (left, dots), doublets (center, lines) and triplets (right, triangles). Similar results were achieved for the unsupervised training data set.

The results for doublets and triplets slightly degrade, but still provide EERs of 87% or better. For comparison, the constellation model proposed by [8] produced an EER of approximately 79% using 40% face images.

MP	EER	Face Tree Size	BG Tree Size
Singlet	88.5	49.4	4.6
Doublet	89.9	49.7	2.8
Triplet	87.6	19.0	0.1

Table 4. Results for Face 1999 data set with only 40% of training images containing faces: EER, average number of unique features in trees for face images and average number of unique features in trees for background images.

5.3. Discovering Objects

In our final test, we demonstrate the ability of our algorithm to build coherent trees from unstructured data. We captured 18 images with random backgrounds containing three objects. The objects include a rotating globe, a woman's face and a deformable t-shirt. Each image containing between 0 and 3 objects. A doublet model was built for the images, and all trees with a probability greater than 0.5 were found ($\epsilon = 10^5 p(O)$). Next, all trees which shared at least 20% common words for their doublets were grouped together. Without supervision, 3 groups were found that corresponded to the three objects. Furthermore, every occurrence of the objects was found, even though the objects deformed (face and t-shirt) and pairs of specific instances of the object may not share features (rotating globe). Half the images in the data set are shown in Figure 9, with the objects labeled by color.

(a)

Figure 7. Example results for Leaves 1999 data set. Largest tree found is shown in red for singlets (left, dots), doublets (center, lines) and triplets (right, triangles).

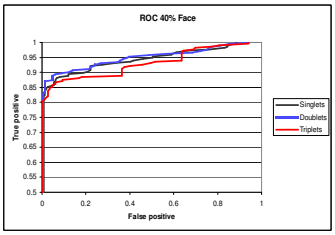


Figure 8. ROC curves for Face 1999 data sets when the training data set only contains 40% faces for singlets, doublets and triplets.

6. Discussion

A drawback of our approach is its reliance on the repeatability of interest point detectors. Without additional information, interest points rely on low-level information for detection. This can lead to unreliable results for many object categories, such as bicycles, cars, airplanes, etc. One

Figure 9. Objects discovered by grouping trees that shared similar words. Groups are shown by color, globe (red), face (green) and t-shirt (blue).

possible extension to our approach is to search for missing features in their predicted locations.

Our approach relies on local information to discriminate between objects. It is possible for false positives to be found in scenes with repetitive textures, in which large trees can be constructed using repeating words. This is especially a concern for less discriminative matching primitives such as singlets.

7. Conclusion

In this paper, we presented a local feature-based model for object recognition. Our bi-gram model consists of transition probabilities between neighboring matching primitives. Several types of matching primitives were explored, including feature singlets, doublets and triplets. Objects are found by constructing trees of matching primitives that are likely given the learned model.

Results show the model is able to generate large trees of matching primitives corresponding to objects across a variety of objects. Object models can also be found using unsupervised learning, with only a small reduction in matching accuracy. Finally, we demonstrated the ability of the algorithm to discover objects unsupervised, even with deforming and rotating objects.

References

[1] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, pages 510–517, 2005. 5

[2] P. Brown, V. Pietra, and P. DeSouza. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992. 1

[3] G. Carneiro and D. Lowe. Sparse flexible models of local features. In *Proceedings of European Conference on Computer Vision*, pages 29–43, 2006. 1, 6

[4] D. Comanicu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 2002. 3

[5] D. Crandall, P. Felzenszwalb, and D. Huttenlocher. Spatial priors for part-based recognition using statistical models. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, pages 10–17, 2005. 1, 6

[6] D. Crandall and D. Huttenlocher. Weakly supervised learning of part-based spatial models for visual object recognition. In *Proceedings of European Conference on Computer Vision*, pages 16–29, 2006. 1

[7] P. Felzenszwalb. Representation and detection of deformable shapes. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 27(2):208–220, 2005. 1

[8] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, 2003. 1, 6, 7

[9] C. J. Harris and M. Stephens. A combined corner and edge detector. In *Proc. 4th Alvey Vision Conferences*, pages 147–151, 1998. 2

[10] T. Kadir and M. Brady. Scale, saliency and image description. *International Journal of Computer Vision*, 30(2):83–105, 2001. 2

[11] S. Lazebnik, C. Schmid, and J. Ponce. Sparse texture representation using affine-invariant neighborhoods. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, 2003. 1, 2

[12] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. 1, 2, 5

[13] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of British Machine Vision Conference*, pages 384–393, 2002. 2

[14] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004. 2

[15] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Proceedings of Computer Vision and Pattern Recognition*, 2006. 1, 2

[16] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Object modeling and recognition using local affine-invariant image descriptors and multi-view spatial contraints. *International Journal of Computer Vision*, 2004. 1

[17] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *IEEE Proceedings of International Conference on Computer Vision*, 2005. 1, 2

[18] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *IEEE Proceedings of International Conference on Computer Vision*, 2003. 2

[19] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *Proceedings of European Conference on Computer Vision*, pages 18–32, 2000. 1