Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting

Yann LeCun, Fu Jie Huang,
The Courant Institute, New York University
715 Broadway, New York, NY 10003, USA
http://yann.lecun.com

Léon Bottou
NEC Labs America,
4 Independence Way, Princeton, NJ 08540
http://leon.bottou.org

Abstract

We assess the applicability of several popular learning methods for the problem of recognizing generic visual categories with invariance to pose, lighting, and surrounding clutter. A large dataset comprising stereo image pairs of 50 uniform-colored toys under 36 azimuths, 9 elevations, and 6 lighting conditions was collected (for a total of 194,400 individual images). The objects were 10 instances of 5 generic categories: four-legged animals, human figures, airplanes, trucks, and cars. Five instances of each category were used for training, and the other five for testing. Low-resolution grayscale images of the objects with various amounts of variability and surrounding clutter were used for training and testing. Nearest Neighbor methods, Support Vector Machines, and Convolutional Networks, operating on raw pixels or on PCA-derived features were tested. Test error rates for unseen object instances placed on uniform backgrounds were around 13% for SVM and 7% for Convolutional Nets. On a segmentation/recognition task with highly cluttered images, SVM proved impractical, while Convolutional nets yielded 16/7% error. A real-time version of the system was implemented that can detect and classify objects in natural scenes at around 10 frames per second.

1. Introduction

The recognition of generic object categories with invariance to pose, lighting, diverse backgrounds, and the presence of clutter is one of the major challenges of Computer Vision. While there have been attempts to detect and recognize objects in natural scenes using a variety of clues, such as color, texture, the detection of distinctive local features, and the use of separately acquired 3D models, very few authors have attacked the problem of detecting and recognizing 3D objects in images primarily from the shape information.

Even fewer authors have attacked the problem of recognizing *generic categories*, such as cars, trucks, airplanes, human figures, or four-legged animals purely from shape information. The dearth of work in this area is due in part to the difficulty of the problem, and in large part to the non-availability of a dataset with sufficient size and diversity to carry out meaningful experiments.

The first part of this paper describes the NORB dataset, a large image dataset comprising 97,200 stereo image pairs of 50 objects belonging to 5 generic categories (four-legged animals, human figures, airplanes, trucks, and cars) under 9 different elevations, 36 azimuths, and 6 lighting conditions. The raw images were used to generate very large sets of greyscale stereo pairs where the objects appear at variable location, scale, image-plane angles, brightness, and contrast, on top of background clutter, and distractor objects.

The second part of the paper reports results of generic shape recognition using popular image classification methods operating on various input representations. The classifiers were trained on five instances of each category (for all azimuths, elevations, and lightings) and tested on the five remaining instances. Results of simultaneous detection and recognition with Convolutional Nets are also reported.

The main purpose of this paper is not to introduce new recognition methods, but rather to (1) describe the largest publicly available dataset for generic object recognition; (2) report baseline performance with standard method on this dataset; (3) explore how different classes of methods fare when the number of input variables is in the tens of thousands, and the number of examples in the hundreds of thousands; (4) compare the performance of methods based on global template matching such as K-Nearest Neighbors and Support Vector Machines, and those based on local feature extraction such as Convolutional Nets, when intra-class variabilities involve highly complex transformations (pose and lighting); (5) assess the performance of template-based methods when the size of the problem is at the upper limit of their practicality; (6) measure to what extent the various learning architectures can learn invariance to 3D pose and lighting, and can deal with the variabilities of natural images; (7) determine whether trainable classifiers can take advantage of binocular inputs.

2. The NORB Dataset

Many object detection and recognition systems described in the literature have (wisely) relied on many different non-shape related clues and various assumptions to achieve their goal. Authors have advocated the use of color, texture, and contours for image indexing applications [8], the detection of distinctive local features [20, 26, 25, 23], the use of global appearance templates [11, 10, 19], the extraction of silhouettes and edge information [14, 22, 8, 4, 19] and the use of pose-invariant feature histograms [9, 5, 1]. Conversely, learning-based methods operating on raw pixels or low-level local features have been quite successful for such applications as face detection [24, 18, 12, 7, 25, 21], but they have yet to be applied successfully to shape-based, pose-invariant object recognition. One of the central questions addressed in this paper is how methods based on global templates and methods based on local features compare on invariant shape classification tasks.

In the NORB dataset, the only useful and reliable clue is the shape of the object, while all the other parameters that affect the appearance are subject to variation, or are designed to contain no useful clue. Parameters that are subject to variation are: viewing angles (pose), lighting condition, position in the image plane, scale, image-plane rotation, surrounding objects, background texture, contrast, luminance, and camera settings (gain and white balance). Potential clues whose impact was eliminated include: color (all images were grayscale), and object texture (objects were painted with a uniform color). For specific object recognition tasks, the color and texture information may be helpful, but for generic shape recognition tasks the color and texture information are distractions rather than useful clues. The image acquisition setup was deliberately designed to reflect real imaging situations. By preserving natural variabilities and eliminating irrelevant clues and systematic biases, our aim was to produce a benchmark in which no hidden regularity can be used, which would unfairly advantage some methods over others.

While several datasets of object images have been made available in the past [11, 22, 19], NORB is considerably larger than those datasets, and offers more variability, stereo pairs, and the ability to composite the objects and their cast shadows onto diverse backgrounds.

Ultimately, practical object recognition systems will have to be trained on natural images. The value of the present approach is to allow systematic objective comparisons shape classification methods, as well as a way of assessing their invariant properties, and the number of examples required to train them.

2.1. Data Collection

The image acquisition system was composed of a turntable on which object were placed, two Hitachi KP-D20AU CCD cameras mounted on a swiveling arm, and four studio lights with bounce umbrellas. The angle of the turntable, the elevations of the camera arm, and the intensity of the lights were all under computer control. The cameras were 41cm away from the objects (roughly arm length) and 7.5cm apart from each other (roughly the distance between the two eyes in humans). The lenses' focal length was set around 16mm. The turntable was 70cm in diameter and had a uniform medium gray color. The lights were placed at various fixed locations and distances around the object.

We collected images of 50 different toys shown in figure 1. The collection consists of 10 instances of 5 generic categories: four-legged animals, human figures, airplanes, trucks, and cars. All the objects were painted with a uniform bright green. The uniform color ensured that all irrelevant color and texture information was eliminated. 1,944 stereo pairs were collected for each object instance: 9 elevations (30, 35, 40, 45, 50, 55, 60, 65, and 70 degrees from the horizontal), 36 azimuths (from 0 to 350° every 10°), and 6 lighting conditions (various on-off combinations of the four lights). A total of 194,400 RGB images at 640×480 resolution were collected (5 categories, 10 instances, 9 elevations, 36 azimuths, 6 lightings, 2 cameras) for a total of 179GB of raw data. Note that each object instance was placed in a different initial pose, therefore "0 degree angle" may mean "facing left" for one instance of an animal, and "facing 30 degree right" for another instance.

2.2. Processing

Training and testing samples were generated so as to carefully remove (or avoid) any potential bias in the data that might make the task easier than it would be in realistic situations. The object masks and their cast shadows were extracted from the raw images. A scaling factor was determined for each of the 50 object instances by computing the bounding box of the union of all the object masks for all the images of that instance. The scaling factor was chosen such that the largest dimension of the bounding box was 80 pixels. This removed the most obvious systematic bias caused by the variety of sizes of the objects (e.g. most airplanes were larger than most human figures in absolute terms). The segmented and normalized objects were then composited (with their cast shadows) in the center of various 96×96 pixel background images. In some experiments, the locations, scales, image-plane angle, brightness, and contrast were randomly perturbed during the compositing process.

2.3. Datasets

Experiments were conducted with four datasets generated from the normalized object images. The first two datasets were for pure categorization experiments (a somewhat unrealistic task), while the last two were for simultaneous detection/segmentation/recognition experiments.

All datasets used 5 instances of each category for training and the 5 remaining instances for testing. In the *normalized* dataset, 972 images of each instance were used: 9 elevations, 18 azimuths (0 to 360° every 20°), and 6 illuminations, for a total of 24,300 training samples and 24,300 test samples. In the various *jittered* datasets, each of the 972 images of each instance were used to generate additional examples by randomly perturbing the position ([-3, +3] pixels), scale (ratio in [0.8, 1.1]), image-plane angle ([-5, 5] degrees), brightness ([-20, 20] shifts of gray levels), contrast ([0.8, 1.3] gain) of the objects during the compositing process. Ten drawings of these random parameters were drawn to generate training sets, and one or two drawings to generate test sets.

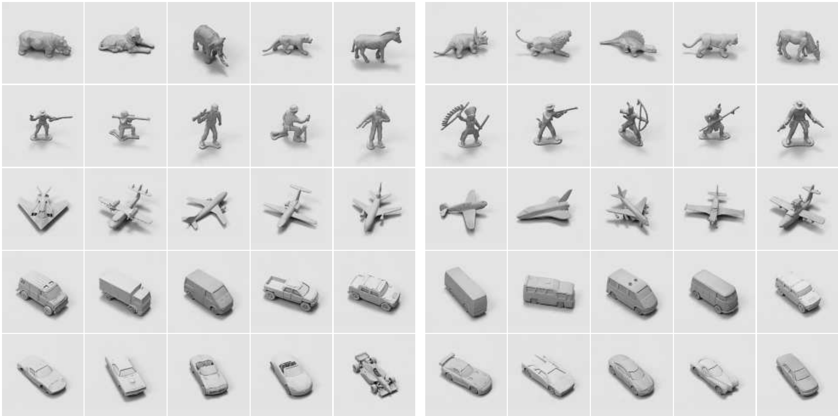

Figure 1. The 50 object instances in the NORB dataset. The left side contains the training instances and the right side the testing instances for each of the 5 categories.

In the *textured* and *cluttered* datasets, the objects were placed on randomly picked background images. In those experiments, a 6-th category was added: background images with no objects (results are reported for this 6-way classification). In the *textured* set, the backgrounds were placed at a fixed disparity, akin to a back wall orthogonal to the camera axis at a fixed distance. In the *cluttered* datasets, the disparities were adjusted and randomly picked so that the objects appeared placed on highly textured horizontal surfaces at small random distance from that surface. In addition, a randomly picked "distractor" object from the training set was placed at the periphery of the image.

- *normalized-uniform set*: 5 classes, centered, unperturbed objects on uniform backgrounds. 24,300 training samples, 24,300 testing samples. See figure 1.
- *jittered-uniform set*: 5 classes, random perturbations, uniform backgrounds. 243,000 training samples (10 drawings) and 24,300 test samples (1 drawing)
- *jittered-textured set*: 6 classes (including one background class) random perturbation, natural background textures at fixed disparity. 291,600 training samples (10 drawings), 58,320 testing samples (2 drawings). See figure 2.
- *jittered-cluttered set*: 6 classes (including one background class), random perturbation, highly cluttered background images at random disparities, and randomly placed distractor objects around the periphery. 291,600 training samples (10 drawings), 58,320 testing samples (2 drawings). See figure 2.

Occlusions of the central object by the distractor occur occasionally, as can be seen in figure 2. Most experiments were performed in binocular mode (using left and right images), but some were performed in monocular mode. In monocular experiments, the training set and test set were composed of all left and right images used in the corresponding binocular experiment. Therefore, while the number of training samples was twice higher, the total amount of training data was identical. Examples from the *jittered-textured and jittered-cluttered training set* are shown in figure 2.

3. Experiments

The following classifiers were tested on raw image pairs from the *normalized-uniform* dataset: linear classifier, K-Nearest Neighbor (Euclidean distance), pairwise Support Vector Machines with Gaussian kernels, and Convolutional Networks [7]. With 18,432 input variables and 24,300 samples, this dataset is at the upper limit of practicality for template-matching-based methods such as K-NN and SVM (in fact, special algorithms had to be implemented to make them practical). The K-Nearest Neighbor and SVM methods were also applied to 95-dimensional vectors of PCA coefficients extracted from the $2 \times 96 \times 96$ binocular training images. All the methods were also applied to Laplacian-filtered versions of the images, but the results were uniformly worse than with raw images and are not reported.

The Convolutional Network was trained and tested on the *normalized-uniform* dataset, as well as on the *jittered-uniform* and *jittered-textured* datasets. The jittered training sets were much too large to be handled by the K-NN and SVM methods within reasonable limits of CPU time and memory requirements. In the following sections, we give brief descriptions of the methods employed. All the algorithms were implemented with the Lush environment [3]. The SVM implementation used the `Torch` library [6] interfaced to Lush, while the Convolutional Net implementation used Lush's `gblearn2` package.

3.1. Principal Component Analysis

Computing the Principal Components of the dataset for the PCA-based K-NN and SVM was a major challenge because it was impossible to manipulate (let alone diagonalize) the 18,432×18,432 covariance matrix ($2 \times 96 \times 96$

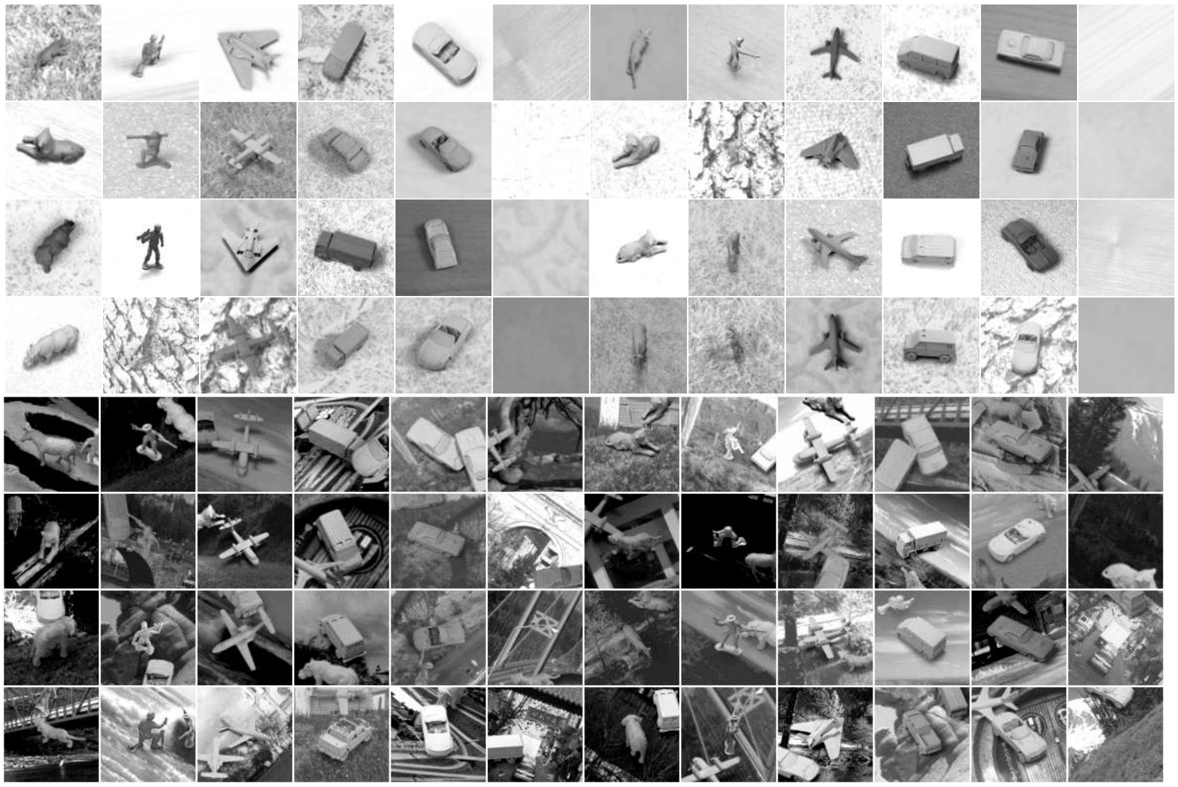

Figure 2. Some of the 291,600 examples from the *jittered-textured* training set (top 4 rows), and from the *jittered-cluttered* training set (bottom 4 rows). (left camera images).

squared). Fortunately, following [13], we can compute the principal direction of a centered cloud of points (x_i) by finding two cluster centroids that are symmetric with respect to the origin: we must find a vector u that minimizes $\sum_i \min\left((x_i-u)^2,(x_i+u)^2\right)$. A quick solution is obtained with online (stochastic) algorithms as discussed in [2] in the context of the K-Means algorithm. Repeated applications of this method, with projections on the complementary space spanned by the previously obtained directions, yield the first 100 principal components in a few CPU hours. The first 29 components thus obtained (the left camera portion) are shown in figure 3. The first 95 principal components were used in the experiments.

3.2. K-Nearest Neighbors (with Euclidean Distance)

Because running the K-Nearest Neighbors algorithm with 24,300 reference images in dimension 18,432 is prohibitively expensive, we precomputed the distances of a few representative images A_k to all the other reference images X_i. By triangular inequality, the distances between a query image X and all the reference image X_i is bounded below by $\mathrm{Max}_k\,|d(X,A_k)-d(A_k,X_i)|$. These can be used to choose which distances should be computed first, and to avoid computing distances that are known to be higher than those of the currently selected reference points [17]. Experiments were conducted for values of K up to 18, but the best results were obtained for $K = 1$. We also applied K-NN to the 95-dimensional PCA-derived feature vectors.

3.3. Pairwise Support Vector Machine (SVM)

We applied the SVM method with Gaussian kernels to the raw images of the *normalized-uniform* dataset, but failed to obtain convergence in manageable time due to the overwhelming dimension, the number of training samples, and the task complexity. We resorted to using the 95-dimensional, PCA-derived feature vectors, as well as subsampled, monocular versions of the images at 48×48 pixels and 32×32 resolutions.

Ten SVMs were independently trained to classify one class versus one other class (pairwise classifiers). This greatly reduces the number of samples that must be examined by each SVM over the more traditional approach of classifying one class versus all others. During testing, the sample is sent to all 10 classifiers. Each classifier "votes" for one of its attributed categories. The category with the largest number of votes wins. The number of support vectors per classifier were between 800 and

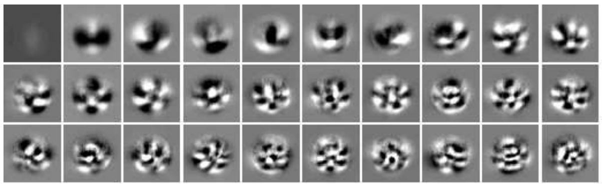

Figure 3. The average image and the first 29 principal eigenvectors of the *normalized-uniform* training set (only the left camera portions of the vectors are shown).

Classification				
exp#	Classifier	Input	Dataset	Test Error
1.0	Linear	raw 2x96x96	norm-unif	30.2%
1.1	K-NN (K=1)	raw 2x96x96	norm-unif	18.4 %
1.2	K-NN (K=1)	PCA 95	norm-unif	16.6%
1.3	SVM Gauss	raw 2x96x96	norm-unif	N.C.
1.4	SVM Gauss	raw 1x48x48	norm-unif	13.9%
1.5	SVM Gauss	raw 1x32x32	norm-unif	12.6%
1.6	SVM Gauss	PCA 95	norm-unif	13.3%
1.7	Conv Net 80	raw 2x96x96	norm-unif	6.6%
1.8	Conv Net 100	raw 2x96x96	norm-unif	6.8%
2.0	Linear	raw 2x96x96	jitt-unif	30.6%
2.1	Conv Net 100	raw 2x96x96	jitt-unif	7.1%
Detection/Segmentation/Recognition				
exp#	Classifier	Input	Dataset	Test Error
5.1	Conv Net 100	raw 2x96x96	jitt-text	10.6%
6.0	Conv Net 100	raw 2x96x96	jitt-clutt	16.7%
6.2	Conv Net 100	raw 1x96x96	jitt-clutt	39.9%

Table 1. Recognition results. "raw 2x96x96" indicates raw binocular images, "raw 1x96x96" indicates raw monocular images, "PCA-95" indicates a vector of 95 PCA-derived features. "norm-unif" refers to the normalized-uniform dataset, "jitt-unif" to the jittered-uniform dataset, "jitt-text" to the jittered-textured dataset, and "jitt-clutt" to the jittered-cluttered dataset.

2000 on PCA-derived inputs (roughly 2×10^6 flops to classify one sample), and between 2000 and 3000 on $32{\times}32$ raw images (roughly 30×10^6 flops to classify one sample). SVMs could not be trained on the jittered datasets because of the prohibitive size of the training set.

3.4. Convolutional Network

Convolutional Networks [7] have been used with great success in various image recognition applications, such as handwriting recognition and face detection. The reader is refered to the above reference for a general discussion of Convolutional Nets. Convolutional Nets use a succession of layers of trainable convolutions and spatial subsampling interspersed with sigmoid non-linearities to extract features of increasingly large receptive fields, increasing complexity, and increasing robustness to irrelevant variabilities of the inputs.

A six-layer net, shown in figure 5, was used in the experiments reported here. The layers are respectively named C1, S2, C3, S4, C5, and output. The C letter indicates a convolutional layer, and the S layer a subsampling layer. C1 has 8 feature maps and uses $5{\times}5$ convolution kernels. The first 2 maps take input from the left image, the next two from the right image, and the last 4 from both. S2 is a $4{\times}4$ subsampling layer. C3 has 24 feature maps that use 96 convolution kernels of size $6{\times}6$. Each C3 map takes input from 2 monocular maps and 2 binocular maps on S2, each with a different combination. S4 is a $3{\times}3$ subsampling layer. C5 has a variable number of maps (80 and 100 in the reported results) that combine inputs from all map in S4 through $6{\times}6$ kernels. Finally the output layer takes inputs from all C5 maps. The network has a total of 90,575 trainable parameters. A full propagation through the network requires 3,896,920 multiply-adds.

The network was trained to minimize the mean squared error with a set of target outputs. For 5-class recognition tasks, we used a traditional place code (one unit active, the other inactive), for 6-class detection/recognition tasks, we added a 6-th target configuration with all output units inactive for the background class (no object in the center of the image).

We used a stochastic version of the Levenberg-Marquardt algorithm with diagonal approximation of the Hessian [7], for approximately 250,000 online updates. No significant over-training was observed, and no early stopping was performed. For experiments with monocular data, the left image was duplicated into the right image, or vice versa with equal probability.

4. Results and Discussion

4.1. Results on the normalized-uniform and jittered-uniform datasets

The results are shown in table 1. To our knowledge, these are the first systematic experiments that apply machine learning to shape-based generic object recognition with invariance to pose and lighting. These results are intended as a baseline for future work with the NORB datasets.

The first section of the table gives results on the *normalized-uniform* database, a somewhat unrealistic setting that assumes that objects can be isolated from their surroundings and have been size-normalized prior to recognition.

The biggest surprise is that brute-force Nearest Neighbor with Euclidean distance on raw pixels works at all, despite the complex variabilities in the data (see lines 1.1 and 1.2 in the table). Naturally, the classification is horribly expensive in memory and CPU time.

Another important lesson is that Gaussian SVM becomes impractical with very large and complex datasets such as NORB. The Gaussian SVM architecture consists of a layer of template matchers, whose prototypes are a subset of the training samples (i.e. a Gaussian bump is placed around each training sample), followed by a layer of linear combinations with learned weights. Since the supervised learning only takes place in the linear layer, the objective function can be made convex (quadratic with box constraints in the case of traditional SVMs). In fact, an often-stated advantage of SVMs is the convexity of their objective function. That property seems to be of little help in our case because of the difficulty of the task (which increases the number of support vectors), the large number of training samples, and the fact that the size of the quadratic program to be solved grows with the square of the number of training samples. We did not obtain convergence on the raw binocular data after several days of CPU time using one of the fastest known implementations of SVMs (from the Torch package [6]). Experiments with reduced resolution monocular images yielded decent results around 13% (see lines 1.4 and 1.5 in table 1). Working from the PCA features yielded similar results (see line 1.6).

Unfortunately, those complexity reductions were still insufficient to allow us experiments with the much larger jittered-textured and jittered-cluttered trainings set. The performance of SVMs on the jittered test set after training on the unjittered training set was predictably abysmal (48% error on PCA features and 34% on raw $1 \times 48 \times 48$) PCA-derived features, which has met with some success in face recognition, only brought marginal improvements over using raw pixels with K-NN (from 18.4 to 16.6% error), and no improvement with SVM.

One should not be misled by the surprisingly good performance of template-based methods on the *normalized-uniform* dataset. This dataset is unrealistically favorable to template-based methods because the lighting conditions are in small numbers (6) and are exactly identical in the training set and the test set. Furthermore, the perfectly uniform backgrounds, perfect object registration, and perfect size normalization are not likely to be possible in realistic object recognition settings.

On the *normalized-uniform* set, convolutional nets reached error rates below 7% with binocular inputs (lines 1.7, and 1.8). The error rate was only mildly affected by jittering the training and test samples (7.1% versus 6.8% for non-jittered). The size of the jittered database was too large to carry out experiments with the template-based methods that would result in meaningful comparisons.

4.2. Results on the jittered-textured and jittered-cluttered datasets

The most challenging task by far was the *jittered-cluttered* dataset, and the less challenging *jittered-textured* dataset, where the classifier must simultaneously detect and recognize objects. The shear size and complexity of these datasets place them above the practical limits of template-based methods, therefore we only report results with Convolutional Nets (lines 5.x and 6.x).

A test error rate of 10.6% on the 6 classes (5 objects plus background) was obtained on the *jittered-textured* dataset. A large proportion of errors were objects classified as background, and cars and space shuttles classified as trucks. A test error rate of 16.7% was obtained on the highly challenging *jittered-cluttered* dataset in binocular mode. An example of the internal state of this network is shown in figure 5. Typical examples of images from the test set and the corresponding answers produced by the system are shown in figure 6.

class	animal	human	plane	truck	car	junk
animal	0.85	0.02	0.01	0.00	0.00	0.11
human	0.01	0.89	0.00	0.00	0.00	0.10
plane	0.01	0.00	0.77	0.02	0.06	0.14
truck	0.03	0.00	0.00	0.84	0.05	0.07
car	0.00	0.00	0.01	0.20	0.69	0.09
junk	0.01	0.02	0.00	0.00	0.00	0.96

Table 2. Confusion matrix on the test set for the binocular convolutional net on the jittered-cluttered database (line 6.0 in the results table). Each row indicates the probability that the system will classify an object of the given category into each of the 6 categories. Most errors are false negatives (objects classified as junk), or cars being classified as trucks.

One significant surprise is the comparatively poor performance of Convolutional Net on the *jittered-cluttered* dataset with monocular inputs (line 6.2): the error rate is 39.9% compared with 16.7% for binocular inputs. This suggests that the binocular network is able to take advantage of the disparity information to help locate the outline of the object and disambiguate the segmentation/classification. In fact, it can be observed on figure 5 that the last 4 feature maps in the first and second layers, which take inputs from both cameras, seem to be estimating features akin to disparity.

5. Conclusion and Outlook

An important goal of this work is to point out the limitations of popular template-based approaches (including SVMs) for classification over very large datasets with complex variabilities. Our results emphasize the crucial importance of trainable local feature extractors for robust and invariant recognition.

A real-time portable demo system was implemented using USB cameras connected to a laptop computer. Convolutional Nets can be scanned over large images very efficiently [7]. Taking advantage of this property, the network is scanned over input images at multiple scales producing likelihood maps for each category. The system can spot and recognize animals, human figures, planes, cars and trucks in natural scenes with high accuracy at a rate of several frames per second. By presenting the input image at multiple scales, the system can detect those objects over a wide range of scales. Examples of output of this system with natural images are shown in figure 4. This figure was generated using the monocular convolutional net trained on the *jitterd-cluttered* database (line 6.2 on the results table). Al-

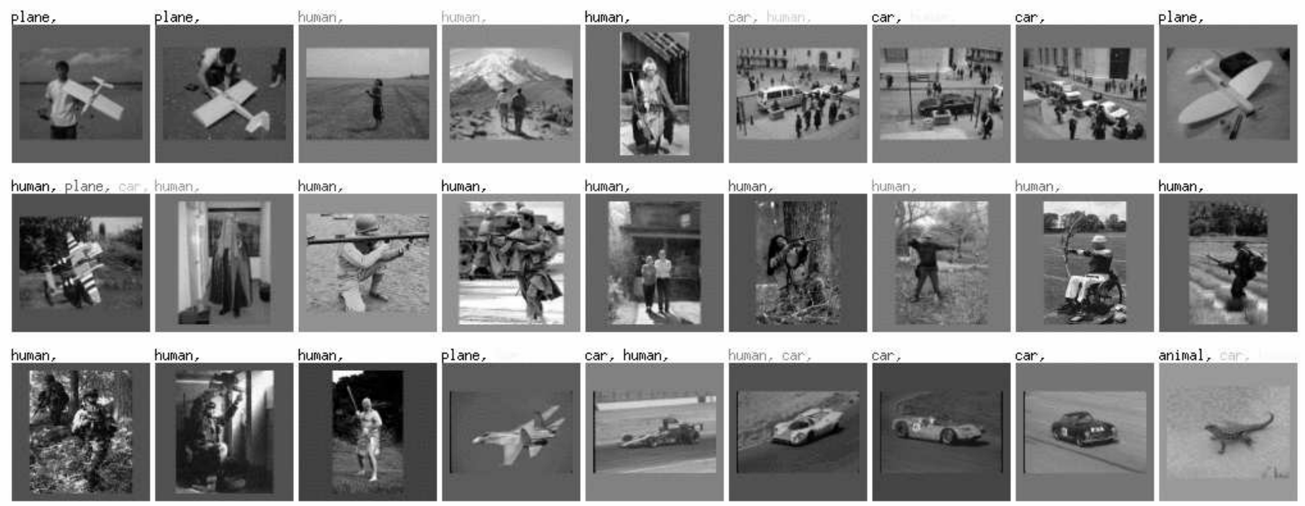

Figure 4. Examples of results on natural images. The list of objects found by the monocular convolutional net is displayed above each sample.

though the raw performance of this network on the database was quite poor, and despite the fact that it was trained only with semi-artificial data, the system can spot most objects in the scenes. The network is applied to the image at two different scales, and is scanned over multiple positions at the large scale. The scores for each class at all scales and positions are combined to produce an overall likelihood of finding an object of the class anywhere in the image. The list of classes whose likelihood exceeds a threshold are shown above each image in the figure. The gray level of the label word is indicative of the likelihood.

The NORB dataset opens the door to large-scale experiments with learning-based approaches to invariant object recognition. This is the first installment in what promises to be a long series of works on the subject. Future work will use trainable classifiers that incorporate explicit models of image formation and geometry.

Acknowledgments We thank Margarita Osadchy, and David Jacobs for useful comments. This work was supported by NSF under ITR grant CCR-0325463.

References

[1] S. Belongie, J. Malik, and J. Puzicha. Matching shapes. In *Proc. of ICCV*, IEEE, 2001.

[2] L. Bottou, Y. Bengio Convergence Properties of the K-Means Algorithm NIPS 7, MIT Press, 1995.

[3] L. Bottou, Y. LeCun Lush Reference Manual `http://lush.sf.net`

[4] O. Carmichael, M. Hebert Object Recognition by a Cascade of Edge Probes. *Proc.e British Mach. Vision Conf.*, 2002

[5] O. Chapelle, P. Haffner, and V. Vapnik, SVMs for Histogram-Based Image Classification, IEEE Trans. Neural Networks, 1999.

[6] R. Collobert, S. Bengio, and J. Mariethoz Torch: a modular machine learning software library. Technical Report IDIAP-RR 02-46, IDIAP, 2002.

[7] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio. Gradient-Based Learning Applied to Document Recognition *Proc. IEEE*, Nov 1998.

[8] J. Malik, S. Belongie, T. Leung, and J. Shi Contour and Texture Analysis for Image Segmentation. *Int. J. of Comp. Vision*, 2001.

[9] B. Mel SEEMORE:Combining color, shape, and texture histogramming in a neuraly-inspired approach to visual object recognition. *Neural Computation*, 9:777-804, 1997.

[10] B. Moghaddam, A. Pentland. Probabilistic Visual Learning for Object Detection. *ICCV*, IEEE, June 1995.

[11] H. Murase and S. Nayar. Visual learning and recognition of 3D objects from appearance. *Int. J. of Comp. Vision*, 14(1):5–24, 1995.

[12] E. Osuna, R. Freund , F. Girosi. Training Support Vector Machines: an Application to Face Detection. *Proc. of CVPR*, Puerto Rico. IEEE, 1997.

[13] M. Partridge, R. Calvo. “Fast Dimensionality Reduction and Simple PCA,” *Intelligent Data Analysis* Vol. 2, No. 3.

[14] J. Ponce, M. Cepeda, S. Pae, S. Sullivan. “Shape models and object recognition.” In D.A. Forsyth et al., editor, *Shape, Contour and Grouping in Computer Vision*. Springer, 1999.

[15] M. Pontil, A. Verri. “Support Vector Machines for 3-D Object Recognition,” *IEEE Trans. Patt. Anal. Machine Intell.* Vol. 20, 637-646, 1998.

[16] S. Agarwal, and D. Roth “Learning a Sparse Representation for Object Detection” *ECCV’02*, May 2002

[17] S. Roweis personal communication, 2003

[18] H.A. Rowley, S. Baluja, T. Kanade. Neural network-based face detection. *IEEE Trans. Patt. Anal. Mach. Intell.*, 20(1):23–38, January 1998.

[19] B. Leibe, and B. Schiele. “Analyzing Appearance and Contour Based Methods for Object Categorization.”, *CVPR*, IEEE, 2003.

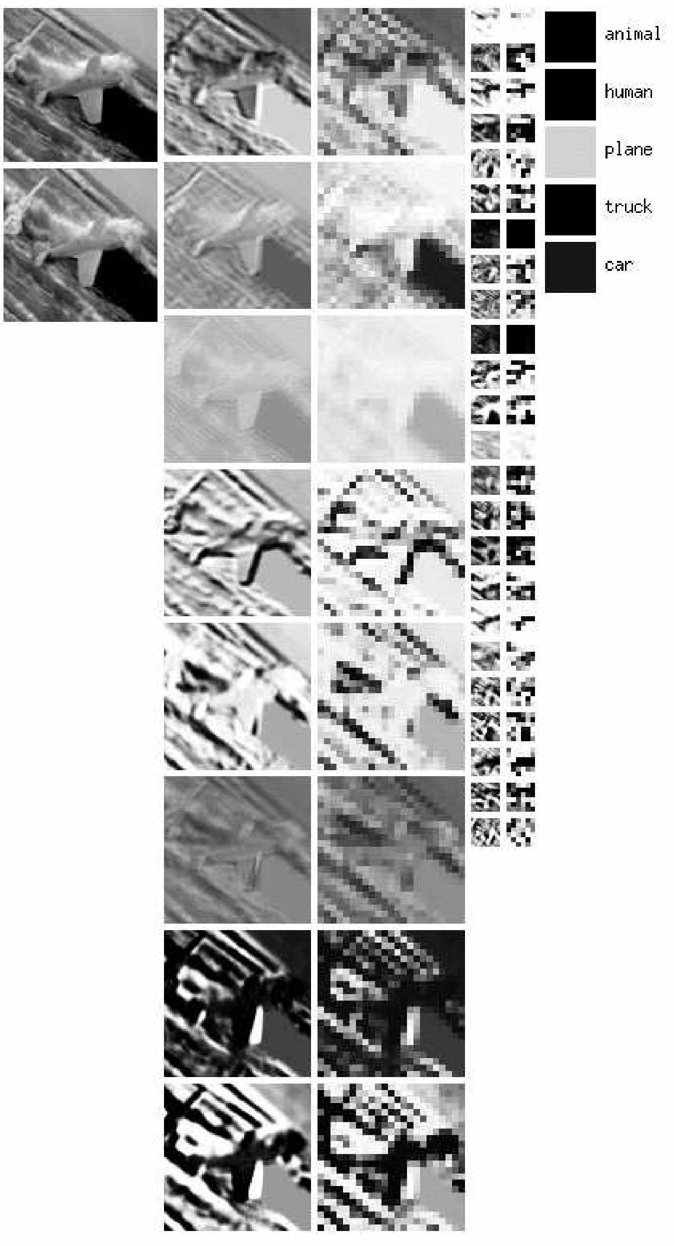

Figure 5. Internal state of the Convolutional Network for an image pair from the *jittered-textured* dataset. From left to right: input (left and right images), C1, S2, C3, S4, and output. layer C5 was omitted. The 4 topmost feature maps of C1 and S2 are monocular, while the 4 bottom ones are binocular.

Figure 6. Examples from the *jittered-cluttered* test set with labels produced by the binocular convolutional net (line 6.0). The labels above each image indicate the system's first choice and second choice ("junk" means no object was found). If the first choice is erroneous, the correct class label is displayed in brackets. This is a typical set of examples where most confusions occur between trucks and cars.

[20] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. Patt. Anal. Mach. Intell.*, 19(5):530–535, May 1997.

[21] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *CVPR*, IEEE, 2000.

[22] A. Selinger, R. Nelson. "Appearance-Based Object Recognition Using Multiple Views," *CVPR*, IEEE, 2001.

[23] S. Ullman, M. Vidal-Naquet, and E. Sali. "Visual features of intermediate complexity and their use in classification", *Nature Neuroscience*, 5(7), 2002.

[24] R. Vaillant, C. Monrocq, and Y. LeCun. Original approach for the localisation of objects in images. *IEE Proc. on Vision, Image, and Signal Proc.*, 141(4):245–250, August 1994.

[25] P. Viola, M. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. *CVPR*, IEEE, 2001.

[26] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *CVPR*, IEEE 2000.