Efficiently Combining Contour and Texture Cues for Object Recognition

Jamie Shotton[†] Andrew Blake[†] Roberto Cipolla[*]
[†]Microsoft Research Cambridge [*]University of Cambridge

Abstract

This paper proposes an efficient fusion of contour and texture cues for image categorization and object detection. Our work confirms and strengthens recent results that combining complementary feature types improves performance. We obtain a similar improvement in accuracy and additionally an improvement in efficiency. We use a boosting algorithm to learn models that use contour and texture features. Our main contributions are (i) the use of dense generic texture features to complement contour fragments, and (ii) a simple feature selection mechanism that includes the computational costs of features in order to learn a run-time efficient model.

Our evaluation on 17 challenging and varied object classes confirms that the synergy of the two feature types performs significantly better than either alone, and that computational efficiency is substantially improved using our feature selection mechanism. An investigation of the boosted features shows a fascinating emergent property: the absence of certain textures often contributes towards object detection. Comparison with recent work shows that performance is state of the art.

1 Introduction

Individually, contour and texture have both been shown to be powerful cues for object recognition [17, 24, 25, 26, 32]. This paper proposes a new efficient synergy of these two cues for image categorization and object detection; see Figure 1.

Our framework uses a boosting algorithm [10] that learns a classifier for object detection by performing feature selection on a heterogeneous pool of features. Learning algorithms typically select features based purely on their classification performance on the training set. We propose a simple additional constraint that incorporates the run-time cost of different features. We show that this *cost-based learning* procedure reduces the run-time cost of the classifier while maintaining the improvement in recognition accuracy gained by combining the contour and texture features.

Related Work. Categorical object recognition is a very active research area [1, 7, 14]. We focus on recent work combining different features. Several papers, e.g. [27, 34], combine different types of local descriptor. Local descriptors, such as SIFT [19] and shape contexts [2], are much closer in descriptive power than the heterogeneous features we propose in this paper. In [8], Fergus *et al*. extend the powerful but computationally expensive constellation model with a basic curve segment descriptor; we use more flexible contour fragments that include both the outline and interior edges of the object. Leibe *et al*. [15] accurately detect pedestrians by post-processing a patch-based recognition model,

BMVC 2008 doi:10.5244/C.22.7

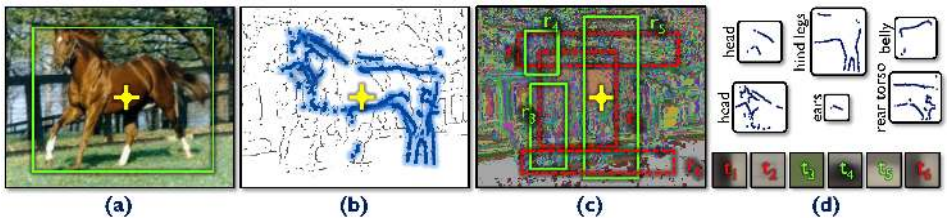

Figure 1: Overview. **(a)** The horse is correctly detected using contour and texture. **(b)** The first six contour fragments in their detected positions on the edge map. **(c)** The first six texture-layout filters with their rectangles on the texton map. Evidence for the horse detection was given by the presence of textures in the solid green boxes and the absence of textures in the dashed red boxes. **(d)** The contour fragments and textons.

but lose the flexibility of parts by matching complete outline contours. Closely related to our technique, Kumar *et al*. [12] combine outline contour and the enclosed texture in pictorial structures, but require tracked video sequences or hand labeled parts; we just require bounding boxes. Another related framework [23] combines contour fragments with sparse raw image patches. These are unlikely to generalize well to categories with repeating textures (e.g. animals). We improve on [23] by using more powerful dense texture-based features and run-time efficiency to drive feature selection. Recently [4] used run-time cost to drive feature selection for simple Haar-like wavelets and discrete Adaboost; our feature selection mechanism is much simpler and should generalize to other feature selection algorithms, not just boosting.

2 Feature Types

In this section, we describe the contour and texture features used in our algorithm. The cue of contour [2, 6, 11, 21, 22, 25] has useful invariance properties, allowing accurate recognition for classes with highly varied surface color and texture, and for images with extreme lighting changes such as silhouetting. However, background edgels that conspire to look similar to the object can lead to false positives. Textural properties [7, 13, 20, 26, 32], such as color and pattern, provide additional semantic cues. While certain textures are clearly useful for recognizing particular object classes (e.g. zebras), an insight in this paper is that the *absence* of certain textures provide evidence for where an object is *not*, e.g. a blue image region is unlikely to be horse. Furthermore, contextual information [28] (e.g. cars often appear above roads) can be straightforwardly exploited using texture-based features [26]. It is intuitively clear that contour and texture are complementary, and this paper confirms and strengthens the results of recent work that this is indeed the case, with results comparable to the state of the art in accuracy, and our new method for run-time cost efficient feature selection giving improvements in efficiency.

2.1 Contour Features

We give a brief overview of the fragments of contour [25] that we use as features. Contour fragments (CFs) are sets of edgels matched to the image using the oriented chamfer distance. The Canny edge map E of the image is pre-processed by computing the distance transform $\mathrm{DT}_E^\tau(\mathbf{x}) = \min(\min_{\mathbf{x}_e \in E} \|\mathbf{x}-\mathbf{x}_e\|_2, \tau)$ (truncated to τ for robustness) and argument distance transform $\mathrm{ADT}_E(\mathbf{x}) = \arg\min_{\mathbf{x}_e \in E} \|\mathbf{x}-\mathbf{x}_e\|_2$, where $\|\cdot\|_2$ is the l_2 norm. The oriented chamfer distance between point sets T (the fragment) and E (the edge map) is computed at 2D translation $\mathbf{x}$ as

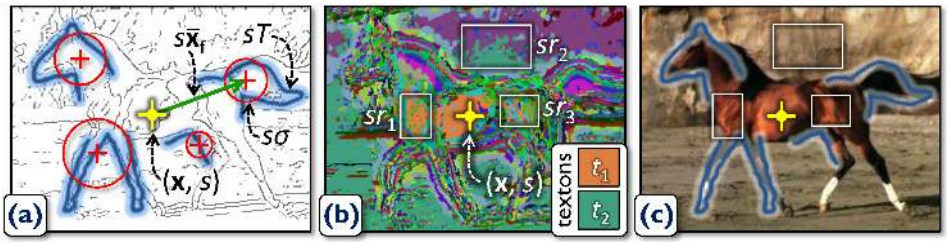

Figure 2: **Features for object detection. (a)** Contour fragments matched against the Canny image edge map. Scaled contour fragments sT are placed relative to the object centroid hypothesis $(\mathbf{x}, s)$. Their expected positions are given by $\mathbf{x} + s\bar{\mathbf{x}}_f$, and have learned spatial uncertainty with radius $s\sigma$. **(b)** Texture-layout filters overlaid on a textonized image. Shapes r_1, r_2 and r_3 are scaled by s, relative to the centroid. Two texton indices t_1 and t_2 are highlighted. See main text. **(c)** Both feature types are used in the combined model. For illustration, only 4 contour fragments and 3 TLFs are shown; in practice we use 100 features.

$$d_\lambda(\mathbf{x}) = (1-\lambda)\cdot d_{\mathrm{c}}(\mathbf{x}) + \lambda \cdot d_{\mathrm{o}}(\mathbf{x}) \;, \tag{1}$$

where λ is the orientation specificity parameter, which interpolates between the distance term

$$d_{\mathrm{c}}(\mathbf{x}) = \frac{1}{\tau|T|} \sum_{\mathbf{x}_{\mathrm{t}} \in T} \mathrm{DT}_E^{\tau}(\mathbf{x}_{\mathrm{t}} + \mathbf{x}) \tag{2}$$

and the orientation term

$$d_{\mathrm{o}}(\mathbf{x}) = \frac{2}{\pi|T|} \sum_{\mathbf{x}_{\mathrm{t}} \in T} |\phi(\mathbf{x}_{\mathrm{t}}) - \phi(\mathrm{ADT}_E(\mathbf{x}_{\mathrm{t}} + \mathbf{x}))| \;. \tag{3}$$

Here $\phi(\mathbf{x})$ gives the orientation of edgel $\mathbf{x}$ modulo π, and $|\phi(\mathbf{x}_1) - \phi(\mathbf{x}_2)|$ gives the smallest circular difference between $\phi(\mathbf{x}_1)$ and $\phi(\mathbf{x}_2)$. The normalization factors in each term mean that $d_\lambda \in [0,1]$. A low value for d_λ indicates a good match.

A codebook $\mathscr{F}$ of contour exemplars is learned from training images using the algorithm in [25]. This selects a set of class-specific fragments (see Figure 1d) and places them in a star-shaped spatial arrangement about the object centroid (Figure 2a). Each resulting exemplar is written $F = (\bar{T}, \bar{\mathbf{x}}_{\mathrm{f}}, \sigma)$, which consists of a set of edgels $\bar{T}$, the expected offset $\bar{\mathbf{x}}_{\mathrm{f}}$ from the object centroid, and the spatial uncertainty σ. The exemplars F are defined at object scale 1, and so to match an object at scale s, $\bar{T}$, $\bar{\mathbf{x}}_{\mathrm{f}}$, and σ are scaled up to $s\bar{T}$, $s\bar{\mathbf{x}}_{\mathrm{f}}$, and $s\sigma$.

For object detection, a set of centroid hypotheses are evaluated (see below). To match contour exemplar F to edge map E for hypothesis $(\mathbf{x}, s)$, a weighted search for the best oriented chamfer match is performed as:

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}'} \left(d_\lambda^{(s\bar{T},E)}(\mathbf{x}') + w_{s\sigma}(\|\mathbf{x}' - (\mathbf{x} + s\bar{\mathbf{x}}_{\mathrm{f}})\|_2) \right) , \tag{4}$$

which weights the oriented chamfer distance for scaled fragment $s\bar{T}$, using the truncated quadratic penalty $w_\sigma(x) = \frac{x^2}{\sigma^2}$ if $|x| \le \sigma$ or ∞ otherwise. The feature response of F paired with a learned λ (see below) is then $v_{[F,\lambda]}(\mathbf{x}, s) = d_\lambda^{(s\bar{T},E)}(\mathbf{x}^\star)$.

2.2 Texture Features

We employ the texture-layout filters from [26]. Originally used for semantic image segmentation, we extend them for multi-scale object detection. Texture-layout filters (TLFs) are based on localized patterns of textons [18, 30], a compact discrete image representation. An image is textonized by assigning each pixel to a particular texton, according to

the output of a filter bank. The background of Figure 2(b) is a textonized image where colors represent texton indices, and Figure 1(d) shows examples of the textures that textons can represent. A TLF is a pair (r,t) of rectangle r and texton t.

In our extension to TLFs, the rectangle $r = (\mathbf{r}_{\text{tl}}, \mathbf{r}_{\text{br}})$ is defined (similarly to the CFs) at object scale 1 relative to the object centroid. For object centroid hypothesis $(\mathbf{x}, s)$, the shape is scaled up to $sr = (s\mathbf{r}_{\text{tl}}, s\mathbf{r}_{\text{br}})$. The response of TLF (r,t) for centroid hypothesis $(\mathbf{x}, s)$ is calculated as:

$$v_{[r,t]}(\mathbf{x}, s) = \frac{1}{\text{area}(sr)} \sum_{\mathbf{x}' \in (sr+\mathbf{x})} [\pi_{\mathbf{x}'} = t] \,, \tag{5}$$

where $\pi_{\mathbf{x}'}$ represents the texton at pixel $\mathbf{x}'$. This calculates the proportion of the rectangle that has a particular texton index. Integral histograms are used for efficiency. Normalizing by the scaled area ensures responses are comparable across scales.

Let us illustrate the flexibility of TLFs through three examples in Figure 2(b). Modeling texture and layout is illustrated by TLF (r_1, t_1): the presence of a large proportion of t_1 (the brown horse texture) in scaled rectangle sr_1 gives positive evidence for the horse centroid at $(\mathbf{x}, s)$. Modeling textural *context* is illustrated by TLF (r_2, t_2): a large proportion of t_2 (the background sandy color) in sr_2 gives evidence for the horse centroid. Finally, TLF (r_3, t_2) can be used to look for a *low* proportion of t_2 (the same sandy color) in sr_3. This means that our model can additionally exploit the absence of that texture on the body of the horse as positive evidence for the horse at $(\mathbf{x}, s)$. The practical demonstration in Figure 4 of this intriguing and intuitive emergent property is one of the contributions of our evaluation. These example features are clearly useful for the particular image in Figure 2, and the learning algorithm will select them if the patterns of evidence prove consistent across the training set. We see real examples of TLFs in Figure 1(c,d).

3 Object Detection

We follow the object detection strategy from [25], with minor modifications. This is based on sliding window classification [1, 9, 31], a simple and effective technique for object detection. The probability of object presence is given by $P(\text{obj}_{(\mathbf{x},s)}) = [1 + \exp(-H(\mathbf{x}, s)]^{-1}$, using the output of a classifier $H(\mathbf{x}, s)$. This is calculated at locations $(\mathbf{x}, s)$ on scaled regular grids with spacing $s\Delta$ for scales $s \in \mathscr{S}$. Mean shift [3] is used to select local maxima as the final set of detections.

The boosted classifier $H(\mathbf{x}, s)$ combines heterogeneous features in an additive model by summing the classifications of M weak learners:

$$H(\mathbf{x}, s) = \sum_{m=1}^{M} a_m [v_m(\mathbf{x}, s) > \theta_m] + b_m \,, \tag{6}$$

where a_m and b_m are classification confidence values for weak learner m, and θ_m is a threshold. Feature response v_m is calculated as:

$$v_m(\mathbf{x}, s) = \begin{cases} v_{[F_m, \lambda_m]}(\mathbf{x}, s) & \text{if } m \text{ contour feature} \\ v_{[r_m, t_m]}(\mathbf{x}, s) & \text{if } m \text{ texture feature.} \end{cases} \tag{7}$$

Mean shift [3] is applied to the hypothesized centroid locations weighted by their scaled posterior probabilities $s^2 P(\text{obj}_{(\mathbf{x},s)})$, similarly to [14]. Multiplying by s^2 compensates for the proportionally less dense hypotheses at larger scales. The algorithm models a non-parametric distribution over hypothesis space with a kernel density estimator, efficiently locating modes that form the final set of detections. The density estimate at each

mode is used as a confidence of the detection. For image categorization we take the global maximum.

Learning. We use GentleBoost [10] to learn classifier (6). This takes a set of training examples i each consisting of feature vector $\mathbf{f}_i$ paired with target value $z_i = \pm 1$, and greedily builds classifier $H(\mathbf{x}, s)$. Example i represents a location $(\mathbf{x}_i, s_i)$ in one of the training images, and the corresponding target value z_i specifies the presence ($z_i = +1$) or absence ($z_i = -1$) of an object there. Examples are chosen in the pattern defined in [25], to encourage a strong, localized positive response near the true object centroid and a negative response to background clutter.

The feature vectors $\mathbf{f}_i$ for all examples i can be thought of as a training matrix where columns represents examples i, and rows represent potential contour or texture features. Each contour feature row maps to an exemplar fragment F paired with an orientation specificity λ (1). There are therefore $|\mathscr{F}| \times |\Lambda|$ contour rows (in our experiments, $200 \times 5 = 1000$), with $\mathscr{F}$ the codebook of contour exemplars, and Λ a discrete set for λ. Each texture feature row maps to a TLF (a rectangle r paired with a texton index t). There are $|\mathscr{R}| \times K$ rows (about $100 \times 200 = 20000$), with $\mathscr{R}$ a randomly selected set of candidate rectangles, and K the number of textons. Using decision stumps as weak learners in (6), the boosting algorithm iteratively selects individual rows of the matrix, thereby performing heterogeneous feature selection.

Cost-Based Learning. The two feature types carry very different computational costs: CFs are much more expensive than TLFs. The standard boosting algorithm greedily selects at round m the optimal weak learner h_m (a contour or a texture feature) from the pool of all possible features $\mathscr{H}$ as $h_m = \arg\min_{h \in \mathscr{H}} J_{\text{wse}}[h]$, where the J_{wse} (defined in [29]) is the training set classification performance.

We propose a simple method for run-time efficiency called cost-based learning. This biases the weak learner selection with a cost associated with the feature type. Writing these costs as Q_{c} for contour and Q_{t} for texture (only the ratio matters), the boosting minimization is modified, so that each weak learner is selected using:

$$h_m = \arg\max_{h \in \mathscr{H}} \frac{1}{Q_h} \left(J'_{\text{wse}} - J_{\text{wse}}[h] \right) , \tag{8}$$

where $Q_h \in \{Q_{\text{c}}, Q_{\text{t}}\}$ matches candidate weak learner h, and J'_{wse} denotes the total training error at the previous round. This maximizes the improvement in classification accuracy on the training set per unit cost. The costs need not necessarily represent the precise relative costs of the different feature types, and can simply be used to bias feature selection.

Some related ideas are suggested in [4, 33]. Our cost-based learning is more general, and should apply to any feature selection algorithm with a cost minimization step. For example, with randomized forest classifiers [16] one could apply a similar multiplication by $\frac{1}{Q_h}$ to the information gain criterion used to recursively build the trees. Furthermore, the granularity of cost-based learning can be as fine as the individual features themselves, for example using cost proportional to the number of edgels in the CFs.

4 Evaluation

We present a thorough evaluation on two challenging datasets, comparing with [23, 24]. Ground truth bounding boxes b_{gt} are provided. The object scale is defined as $s = \sqrt{\text{area}(b_{\text{gt}})}$.

The detection algorithm returns a set of confidence-valued scale-space object centroids. We assign a scaled bounding box b_{inf} centered on each detection, with the same

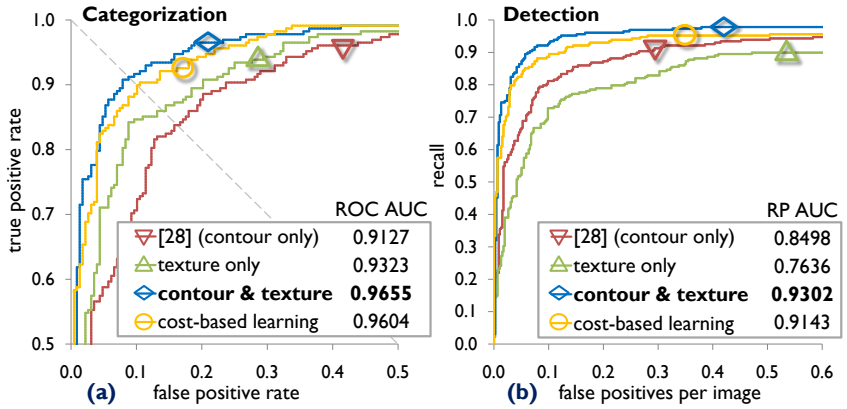

Figure 3: **Comparing contour and texture performance.** **(a)** ROC curves for categorization. **(b)** RFPPI curves for detection. Texture alone is better than contour at categorization, but due to its poor localization, worse at detection. The combination of features is significantly better for both categorization and detection, and substantially better than [25]. By weighting combined feature selection according to the computational cost of the features, cost-based learning enables performance almost as good as standard learning, at a fraction of the cost.

aspect-ratio as the average training bounding box. To be accepted, b_{inf} must overlap with b_{gt}: $\frac{\text{area}(b_{\text{inf}} \cap b_{\text{gt}})}{\text{area}(b_{\text{inf}} \cup b_{\text{gt}})} > 0.5$. Spurious detections of the same object count as false positives.

We used two datasets in our evaluation. (1) The multi-scale Weizmann horse database [25], with background images from Caltech 101 [5]. Images were down-sampled to a maximum 320 pixels width or height. The first 50 horse and background images were used for training, the next 50 as a validation set for optimizing parameters, and a final 228 as the test set. (2) The Graz dataset [23, 24]. This demanding 17 class database includes the relatively easy 'Caltech 4' classes (airplanes, cars (rear), motorbikes, and faces), and some much more challenging classes (see Figure 5). We use the same experimental set-up as [24]. The more limited evaluation in [23] used just the Caltech 4 classes and needed much more training data. Our comparison with this work is therefore only indicative.

4.1 Multi-scale Weizmann Horses

We show in Figure 3 the categorization and detection results comparing (i) the contour only results from [25], (ii) our results using only texture, (iii) our fusion of contour and texture, and (iv) the new cost-based learning criterion for feature selection. The combination of contour and texture features gives significantly superior performance to either individually for both categorization and detection, and significantly improves on the state of the art. Interestingly, texture features are better individually at categorization, while contour features proved better individually at detection, probably because the large spatial extents of TLFs can only give an approximate object localization. The combination appears to use CFs to accurately localize the object, and texture to reject false positives.

The number of features (homogeneous or heterogeneous) is fixed at $M = 100$. The other parameters were set as follows: $\tau = 30$ (2), $K = 200$ textons, $|\mathscr{R}| = 100$ rectangles, $\delta_1 = 0.03$, $\delta_2 = 0.25$, $\gamma_1 = \log 1.1$, $\gamma_2 = \log 1.4$, $\lambda \in \Lambda = \{0, 0.2, \ldots, 1.0\}$ (1), and $\Delta = 0.07$. $|\mathscr{S}| = 6$ test scales were chosen automatically to cover the scale range in the training data. In our unoptimized C# implementation, contour alone took an average of 17.5 seconds per image, texture 4.6s, and the combination without cost-base learning (i.e. $Q_c = Q_t$) 12.5s. Mean shift took 3.4s of these timings. The combination substantially improves

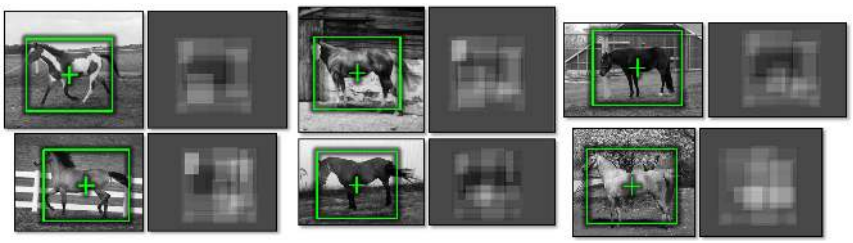

Figure 4: **Results and texture feature visualization.** Six correct detections: the green rectangles are the inferred bounding boxes b_{inf}. The visualization overlays rectangle r for each texture-layout filter, using white when the feature uses texture presence as evidence for the horse, and black when it uses texture absence. Observe a dark region often appears over the body of the horse (though not always). This suggests that the large within-class textural variation prevents boosting from selecting general texture presence features, and so instead it selects features that respond to texture absence, e.g. not green grass or blue sky.

quantitative performance above that achieved by contour features alone, while decreasing the computational cost.

Analysis of Selected Features. Of the 100 features chosen by boosting, 65 were CFs and 35 were TLFs. This suggests that contour is slightly more useful than texture for this dataset, though both play an important role.

Figure 4 shows a few example detections given by the combined detector, and visualizes the TLFs used. The visualization overlays each shape r, using white for texture presence detection or black for texture absence detection.[1] We see from the black regions in Figure 4 that the absence of particular textures on the bodies of the horses contributes to their detections. This makes sense, given the extreme within-class textural variation of horses.

Figure 1(c,d) shows the first six CFs and TLFs selected by boosting. We see examples of CFs that correspond to our notions of 'head', 'hind legs', 'belly', 'ears', and 'rear torso'. Highlighting three particular TLFs, evidence for the object centroid comes from the presence of the 'ear'-like texton t_4 in r_4 (appearance and layout), the presence of 'grass' in r_3 (appearance context), and the absence of the gray color t_2 in r_2.

Cost-Based Learning. We show in Figure 3 the quantitative performance of the combined recognition system, with and without cost-based learning. As before, $M = 100$ weak learners are used. Contour fragments were observed in experiment to be about 40 times more expensive than TLFs. Using this true ratio ($Q_{\text{c}} = 40Q_{\text{t}}$) resulted in only TLFs being chosen, since none of the CFs were at least 40 times better at classification than the TLFs. For a practical demonstration of cost-based learning, we therefore set the relative costs as $Q_{\text{c}} = 5Q_{\text{t}}$. The resulting classifier used 22 contour features and 78 texture features. We see that quantitative performance is slightly reduced, since fewer of the more discriminative, but expensive, contour features are chosen. However, the time per image is reduced to 7.8s per image from 12.5s. Accounting for time taken performing the mean shift, this is a doubling in speed. Although perhaps only a moderate speed increase, there is virtually no additional learning cost and the strong accuracy improvement is substantially maintained.

[1] A weak learner $a[v_{[r,t]}(\mathbf{x},s) > \theta] + b$ with a positive classification confidence a indicates that the presence of texton t (quantified through response $v_{[r,t]}$ in (5)) contributes positively to the object detection. Conversely, a negative a indicates that the absence of t contributes to the object detection.

4.2 Graz

Figure 5 shows results for the Graz dataset. For several very challenging classes we achieve perfect or near perfect categorization and/or detection performance: airplanes, cars (rear), motorbikes, faces, cows (side), cows (front), and cups. For categorization, the classes for which contour and texture features perform better individually are roughly balanced. However, for detection, CFs appear more powerful for most classes, again probably since TLFs are poor at precisely localizing objects. Combining features gives performance that is almost always as good as, and in several cases significantly better than, the performance of contour or texture alone. We highlight the categorization improvements for bikes (side and front), people, cups, and cars ($\frac{2}{3}$ rear), and the detection improvements for bikes (side) and cars (front).

For a few classes, the combined features perform worse than the better of the individual features, such as for detection, horses (front), mugs, and bikes (rear). In these cases, the texture features alone performed poorly, perhaps due to insufficient training data, and have tainted the combined detector. Conversely, for categorization of horses (side and front) and mugs, it is contour that performs worse individually and worsens the combined detector. This shows a limitation of using boosting: its greedy nature does not guarantee optimal feature selection, and so the combination can sometimes get worse. As future work we would like to investigate this effect further and see if different feature selection algorithms can prevent this.

We compare detection performance against [24]. For all but one of the eight classes with the most training data, performance is as good as or better than [24], although their technique does seem to degrade more gracefully with fewer training images. Perhaps for few training examples, the boosting algorithm we employ generalizes less well that theirs, and we postulate that our method would perform better on these classes were more training images provided. We also make an informal comparison with [23], where detection EERs were: airplanes 4.2, cars (rear) 0.0, motorbikes 2.0, and faces 1.0. Since their evaluation used four times more training data for the top three classes, we would expect their performance to be slightly above ours. Despite this, we show significant improvement on their results for motorbikes using only a quarter of their training data.

Feature Types. Shown in Figure 5 (right) are the proportions of contour and texture features used in the combined detector. These proportions are roughly equal for most classes, although certain classes show a significant bias. For motorbikes, faces, mugs, and cups, contour features are selected significantly more frequently. These classes do tend to have very distinctive contours, but less distinctive textures. Conversely, for cars (rear) and bikes (rear), more texture features are selected.

5 Conclusions

We have proposed a new fusion of contour features with dense texture features for multi-scale recognition. Our thorough demonstration on 17 challenging object classes confirms that this combination of contour and texture features can markedly improve results above what either feature type can attain individually, and gives results comparable to or better than the state of the art. The combination also considerably increases the detector speed compared to using CFs alone. We saw how the object detector can exploit both presence and absence of particular textures, and how appearance context is harnessed. A simple cost-based learning mechanism was proposed that maintains high accuracy while substantially reducing run-time cost.

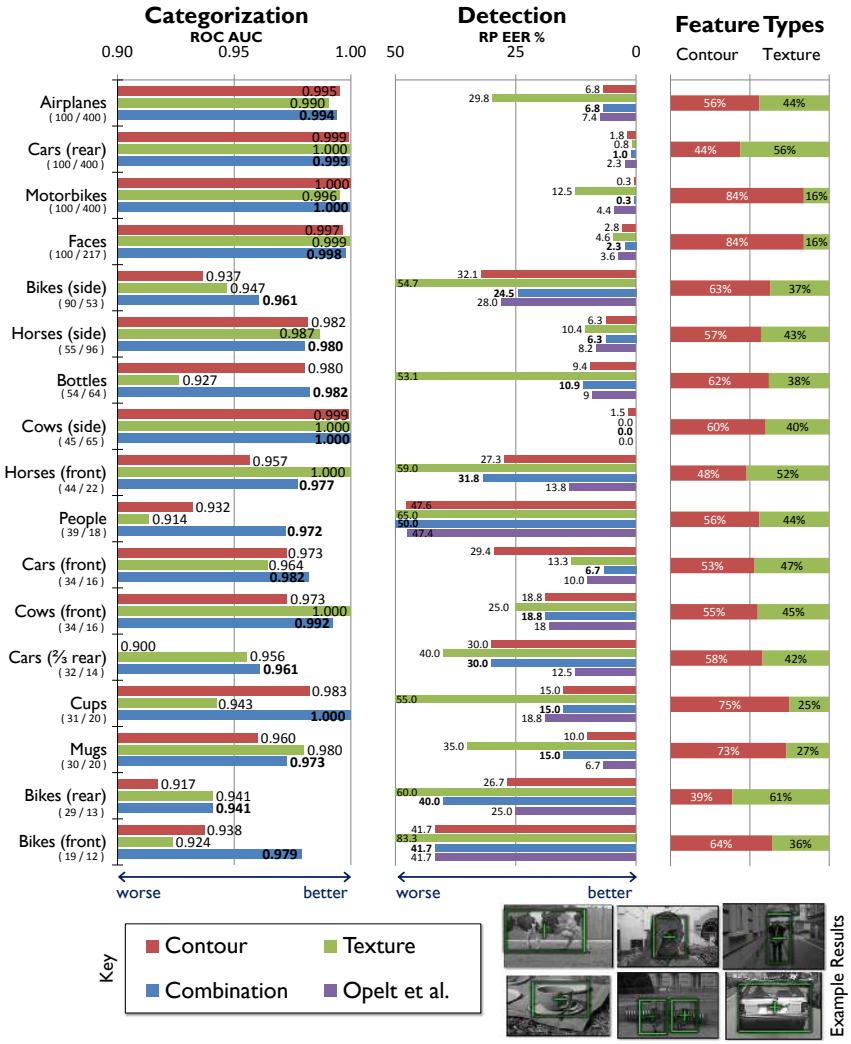

Figure 5: **Results on the Graz dataset.** Performance is compared between only contour, only texture, and our combination of the two. **Left:** categorization performance. **Middle:** detection performance, with comparison to Opelt *et al*. [24]. **Right:** the proportions of contour and texture features used in the combination. Example correct detections are shown bottom right. See text for analysis.

As future work, we aim to investigate different methods of feature combination, for example combining two separately trained classifiers. We would like to apply cost-based learning to other algorithms using finer-grain control over the costs. Finally, we would like to improve the efficiency of the model using a cascade [31] or tree [16].

References

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV*, pages 113–130, 2002.

[2] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *PAMI*, 24(24):509–522, 2002.

[3] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *PAMI*, 24(5), 2002.

[4] P. Dollár, Z. Tu, H. Tao, and S. Belongie. Feature mining for image classification. In *CVPR*, 2007.

[5] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *PAMI*, 28(4):594–611, 2006.

[6] P.F. Felzenszwalb. Learning models for object recognition. In *CVPR*, volume 1, pages 1056–1062, December 2001.

[7] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, volume 2, pages 264–271, June 2003.

[8] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for google images. In *ECCV*, pages 242–256, 2004.

[9] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. Technical Report 5980, INRIA, September 2006.

[10] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28(2):337–407, 2000.

[11] D.M. Gavrila. Pedestrian detection from a moving vehicle. In *ECCV*, pages 37–49. Springer, 2000.

[12] M.P. Kumar, P.H.S. Torr, and A. Zisserman. Extending pictorial structures for object recognition. In *BMVC*, 2004.

[13] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an Implicit Shape Model. In *ECCV'04 Workshop on Statistical Learning in Computer Vision*, May 2004.

[14] B. Leibe and B. Schiele. Scale invariant object categorization using a scale-adaptive mean-shift search. In *DAGM'04: 26th Pattern Recognition Symposium*, pages 145–153, June 2004.

[15] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *CVPR*, pages 1: 878–885, 2005.

[16] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *CVPR*, pages 2:775–781, 2005.

[17] T. Leung. Texton correlation for recognition. In *ECCV*, pages 203–214, 2004.

[18] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, June 2001.

[19] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, November 2004.

[20] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *IJCV*, 43(1):7–27, June 2001.

[21] R.C. Nelson and A. Selinger. A Cubist approach to object recognition. In *ICCV*, pages 614–621, Bombay, India, January 1998.

[22] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *ECCV*, volume 2, pages 575–588, 2006.

[23] A. Opelt, A. Pinz, and A. Zisserman. Fusing shape and appearance information for object category detection. In *BMVC*, 2006.

[24] A. Opelt, A. Pinz, and A. Zisserman. Incremental learning of object detectors using a visual shape alphabet. In *CVPR*, volume 1, pages 3–10, June 2006.

[25] J. Shotton, A. Blake, and R. Cipolla. Multi-scale categoricical object recognition using contour fragments. *PAMI*, To appear 2008.

[26] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, To appear 2008.

[27] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, pages 2: 1470–1477, 2003.

[28] A. Torralba, K.P. Murphy, and W.T. Freeman. Contextual models for object detection using boosted random fields. In *NIPS*, pages 1401–1408, 2005.

[29] A. Torralba, K.P. Murphy, and W.T. Freeman. Sharing features for multiclass and multiview object detection. *PAMI*, 29(5), May 2007.

[30] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *IJCV*, 62(1-2):61–81, 2005.

[31] P. Viola and M.J. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages 1: 511–518, 2001.

[32] J. Winn, A. Criminisi, and T. Minka. Categorization by learned universal visual dictionary. In *ICCV*, pages 2: 1800–1807, 2005.

[33] P. Yin, A. Criminisi, J. Winn, and I. Essa. Tree based classifiers for bilayer video segmentation. In *CVPR*, 2007.

[34] W. Zhang, B. Yu, G.J. Zelinsky, and D. Samaras. Object class recognition using multiple layer boosting with heterogeneous features. In *CVPR*, volume 2, pages 323–330, June 2005.