21st International Conference on Pattern Recognition (ICPR 2012)
November 11-15, 2012. Tsukuba, Japan

Learning to Count with Regression Forest and Structured Labels

Luca Fiaschi
HCI/IWR Heidelberg
luca.fiaschi@iwr.uni-heidelberg.de

Rahul Nair
HCI/IWR Heidelberg
rahul.nair@iwr.uni-heidelberg.de

Ullrich Koethe
HCI/IWR Heidelberg
ullrich.koethe@iwr.uni-heidelberg.de

Fred A. Hamprecht
HCI/IWR Heidelberg
fred.hamprecht@iwr.uni-heidelberg.de

Abstract

Following [Lempitsky and Zisserman, 2010], we seek to count objects by integrating over an object density map that is predicted from an input image. In contrast to that work, we propose to estimate the object density map by averaging over structured, namely patch-wise, predictions. Using an ensemble of randomized regression trees that use dense features as input, we obtain results that are of similar quality, at a fraction of the training time, and with low implementation effort. An open source implementation will be provided in the framework of http://ilastik.org.

1. Introduction

Counting objects in images or video frames is important in many real-world applications including industrial inspection, cytometry, surveying and surveillance. Recent work has shown that object counting can be solved with equal or better accuracy *without* prior object detection or even segmentation [9, 14, 2]. In fact, such an approach is the only one viable in settings of such crowding or such low resolution that detection and segmentation of individuals becomes impracticable. In some instances, a count estimate may also boost the performance of object detectors [13].

An estimate of the object count N_o can either be obtained directly, by mapping from a set of global features to the real line or the integers [3, 7, 10]; or it can be obtained by integrating an estimated density function $F(x)$ over the image domain Ω

$$N_o = \int_\Omega F(x)dx \qquad (1)$$

where $F(x)$ is computed from local features. The latter approach delivers state of the art performance while requiring less training images than global regression methods. In particular, the pioneering work [9] posits $F(x) = \mathbf{c}^T\phi(x)$, for local features $\phi(x)$. The weight vector $\mathbf{c}$ is learned from a training set by solving a quadratic program which minimizes the error between the true and predicted density estimates, integrated over all possible sub-windows. This model works very well, even though the predicted density can be negative in places, and the linear model requires a relatively complex set of features (BoW-SIFT).

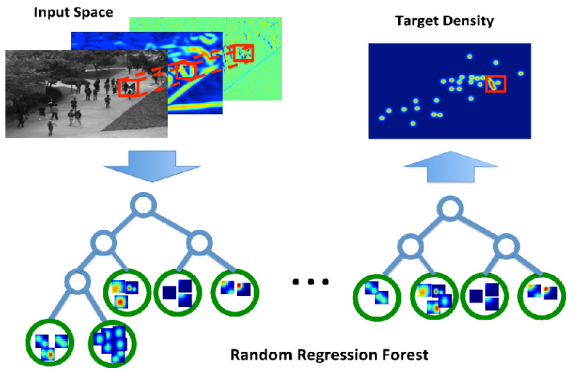

Figure 1. Summary of our framework. A regression random forest learns a mapping between patches in the input feature space and in the target object density space. Overlapping predictions of patches are averaged to obtain a density of objects per pixel.

Our main contribution is a simplification of the original approach, arguably on the conceptual level and that of the implementation effort, and certainly in terms of

computational effort in the learning phase. In particular, we compute dense features by ordinary filter banks; and propose to use a regression forest to predict entire patches of the desired density function. These structured predictions are averaged both across predictors in the ensemble, and across space, akin but not identical to [8]. In addition, we leverage the random forest out-of-bag samples to compute an uncertainty measure on the predicted object density. In section 3, we show that results, on some dataset, compare favorably with the state of the art.

2. Predicting a structured regression target

Our algorithm requires a set of training images $I_i,\ i \in 1, ..., N$, where all objects present in the image must be annotated with one "dot" in the center. The true density function for each pixel $x \in I_i$ is defined as a sum of Gaussian kernels centered on the user annotations:

$$F_i^0(x) = \sum_{\mu \in \mathbf{A}_i} \mathcal{N}(x; \mu, \sigma) \tag{2}$$

where $\mathbf{A}_i$ is the set of all annotations for image I_i and σ is a smoothness parameter. This parameter is fixed at $\sigma = 2.5$ in all reported experiments of the last section, amounting to roughly $1/4$ the objects' size. The results obtained are not very sensitive to the precise choice of σ. Since the basis functions are normalized, the overall number of objects in an image can be computed by equation 1.

Our key observation is that equation 2 represents a smooth function. As a consequence, instead of predicting the density at each location x individually, we incorporate neighborhood information by making predictions for dense overlapping patches. These overlapping predictions are then averaged in order to reduce the single pixel error in the estimate.

Rather than work on the raw images, we first compute a number ν of standard filter bank responses and then learn a nonlinear mapping

$$\mathcal{F} : \mathbf{P}_{in} \rightarrow \mathbf{P}_{out} \tag{3}$$

from a $h \times w$ patch in the input space $\mathbf{P}_{in} \in \mathbb{R}^{h \times w \times \nu}$ to a patch in the output or target space $\mathbf{P}_{out} \in \mathbb{R}^{h' \times w'}$. Given the mapping $\mathcal{F}$, the predicted density estimate per pixel is obtained by averaging all the predicted overlapping patches:

$$\hat{F}(x) = \frac{1}{|\mathcal{P}(x)|} \sum_{\hat{\mathbf{P}}_{out} \in \mathcal{P}(x)} \hat{\mathbf{P}}_{out}(x) \tag{4}$$

where $\mathcal{P}(x)$ is the set of predicted patches $\hat{\mathbf{P}}_{out}$ that have pixel x in their scope. This formula works for any method that can predict not merely a single scalar, but an entire patch at a time. We have opted for regression forest [1, 4] since it offers the advantage of efficient learning and inference (around $\mathcal{O}(n \log n)$ and $\mathcal{O}(\log n)$, respectively for a fully grown tree) and high performance without parameter tweaking. In addition, it naturally leads to a confidence interval (in a loose sense) for the prediction as explained in the following.

2.1. Regression Forest using Structured Labels

A regression random forest is an ensemble of regression trees, each of which associates a continuous prediction with each input. Given a training set of tuples of patches $\{\mathbf{P}_{in}, \mathbf{P}_{out}\}$, we construct the trees on a randomized subset of the training examples, while the rest is kept as out-of-bag. The learning proceeds recursively, by splitting all data $\mathcal{S}_j$ arriving at a node j into a left and right subset $\mathcal{S}_L, \mathcal{S}_R$. The split is chosen by thresholding at a value τ of some simple test function f:

$$\begin{aligned} \mathcal{S}_L &= \{i \in S_j | f(\mathbf{P}_{in}) < \tau\} \\ \mathcal{S}_R &= \mathcal{S}_j \backslash \mathcal{S}_L \end{aligned}$$

At every internal node, several test functions f are randomly selected. We choose the simplest functional form of f, viz. the value of the observation at a specific pixel position and channel index inside a patch. Given a set of test functions, the best combination of test function and threshold value is found by maximizing

$$-\sum_{i \in \mathcal{S}_R} ||\mathbf{P}^i_{out} - \bar{\mathbf{P}}^R_{out}||^2_F - \sum_{i \in \mathcal{S}_L} ||\mathbf{P}^i_{out} - \bar{\mathbf{P}}^L_{out}||^2_F$$

where $\bar{\mathbf{P}}_{out}$ is the average patch response and $||\cdot||_F$ is the Frobenius norm. The recursive splitting ends when a maximum tree depth is reached (see experiments) or when $|S_j|$ is smaller than a given number. As a result of this construction, the variance of all patches associated with a node decreases with increasing tree depth, and the leaves contain clusters of similar patches.

For a new test sample $\mathbf{P}_{in}$, the total response of the forest is the average of the patches stored during training time in the leaves:

$$\hat{\mathbf{P}}_{out} = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} \bar{\mathbf{P}}^l_{out} \tag{5}$$

where $\mathcal{L}$ is the set of leaves reached by $\mathbf{P}_{in}$. It is to note that in contrast to ordinary scalar regression forest, in our approach, each output is now a complete patch. Figure 2 shows a result of the described procedure, and a comparison with [9][1].

[1] Using the implementation kindly provided by the authors, and the suggested parameters $\sigma = 4$, $N = 10$ training images, a dictionary of size 256 and $C = 0.011$.

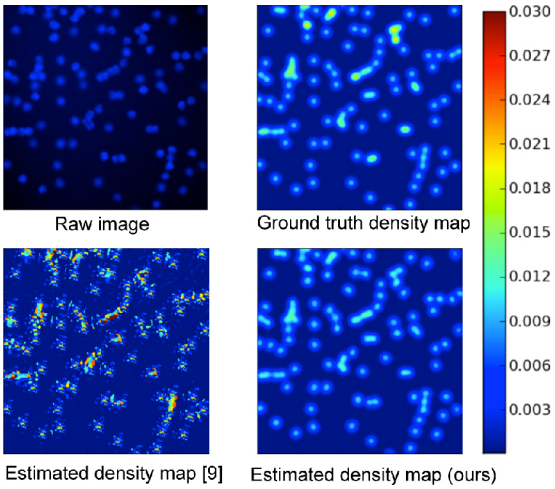

Figure 2. Estimated bacterial density maps. The proposed procedure makes for much smoother estimates. (Colorbar refers to all but the raw image.)

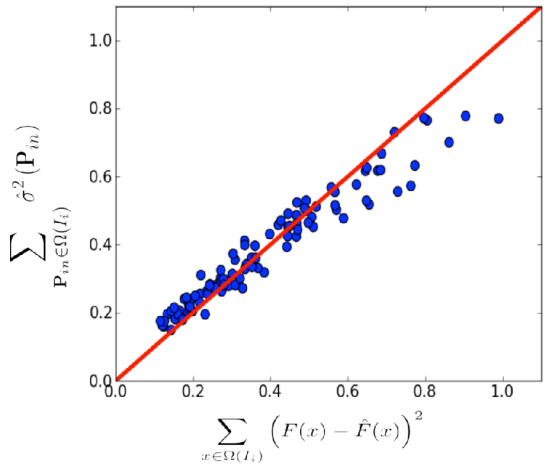

Figure 3. For every image of the bacterial microscopy test dataset, sum of local uncertainties vs. sum of squared local density errors.

2.2. Uncertainty of a Prediction

Scalar quantile regression forests use all in-bag training examples in the leaves [11]. In our multidimensional regression case, we estimate the uncertainty of the predictions using the residual variance of both in-bag and out-of-bag samples in the leaves. Using out-of-bag samples only would seem ideal to obtain an unbiased estimate even in the small-sample case; unfortunately, not all leaves do receive out-of-bag samples. As a compromise, we push all samples from the training set (both in- and out-of-bag) down a tree and compute the total variance for each leaf l as

$$\sigma_l^2 = \frac{1}{|\mathcal{S}_l|} \sum_{i \in \mathcal{S}_l} ||\mathbf{P}^i_{out} - \bar{\mathbf{P}}^l_{out}||_F^2 \qquad (6)$$

For a new test sample $\mathbf{P}_{in}$, we can then assign an uncertainty measure by averaging across all trees where $\mathcal{L}$ is, again, the set of all leaves that sample $\mathbf{P}_{in}$ ends up in:

$$\hat{\sigma}^2(\mathbf{P}_{in}) = \frac{1}{|\mathcal{L}|} \sum_{l \in \mathcal{L}} \sigma_l^2 \qquad (7)$$

Figure 3 shows that this uncertainty is related to the test set error, as desired.

3. Experimental results

In order to compare our algorithm with the state of the art, we conduct our experiments on two publicly available benchmarks. For simplicity, in the following, we use square input and output patches of same size, so $h = w = h' = w'$. An ensemble of 30 trees is used in all experiments, and the minimum split size is set to 20.

3.1. Bacterial cells microscopy images

The dataset [9] is composed of 200 simulated fluorescence microscopy images of cell cultures (171 ± 64 cells on average), with 100 images reserved for training and 100 for validation. We use a random subset of N out of all training images. For every N, we repeat the experiment five times to obtain standard errors. The maximum depth of trees is set to 10, and 3000 patches (or 1000 for N=32) are randomly sampled from each training image, with 30% of them kept out-of-bag for each tree. Following [9], we discard the green and red channels of the raw images, and use the blue channel as well as the following features computed from it: Laplacian of Gaussian, Gaussian gradient magnitude and eigenvalues of the structure tensor at scales 0.8, 1.6, 3.2.

The results are reported in table 1. We also include results for pixel to pixel regression, which emerges as a special case for $h = 1$. Note that even this method uses information from a local neighborhood, through the finite width of the filters whose response is used as features for the prediction. In fact, pixel to pixel regression already performs pretty well, but extension of the regression to use and predict entire patches further improves performance and reduces variability.

Table 1. Mean absolute errors for cell counting on microscopy images [9]

	$N = 2$	$N = 4$	$N = 8$	$N = 32$
Detection + correction [9]	22.6 ± 5.3	16.8 ± 6.5	6.8 ± 1.2	4.9 ± 0.5
Density MESA [9]	5.6 ± 1.5	4.9 ± 0.6	4.9 ± 0.7	3.5 ± 0.2
This work, $h = 1$	9.1 ± 5.5	4.2 ± 0.9	3.5 ± 0.3	3.3 ± 0.3
This work, $h = 5$	7.7 ± 4.6	4.2 ± 1.1	4.4 ± 2.0	3.3 ± 0.2
This work, $h = 7$	$\mathbf{4.8 \pm 1.5}$	$\mathbf{3.8 \pm 0.7}$	$\mathbf{3.4 \pm 0.1}$	$\mathbf{3.2 \pm 0.1}$

Table 2. Mean absolute errors for people counting in surveillance video [2]

	'maximal'	'downscale'	'upscale'	'minimal'
Counting-regression [14]	1.80	2.34	2.52	4.46
Counting-segmentation [14]	**1.53**	1.64	1.84	**1.31**
Density MESA [9]	1.70	**1.28**	**1.59**	2.02
This work, $h = 7$	1.70	2.16	1.61	2.20

3.2. Pedestrians in surveillance video

This data set [2] comprises 2000 video frames from a surveillance camera, along with dotted ground truth (29 ± 9 pedestrians on average). We compare our method with the best results obtained in the recent evaluations [14, 9] following the proposed experimental protocol. Note that the "counting by segmentation" and "counting by regression" [14] methods require a post-processing step to correct for large differences between the estimated counts in consecutive frames. As pre-processing, we subtract from the original images a static background as estimated by a median filter. As in the previous experiment, as features we use the raw image augmented by the filter responses described before, with the addition of a temporal derivative filter. We correct all channels for perspective distortion by simply multiplying their pixel values with the square of the provided camera perspective map. As in [14, 9] we split the data into four different training and test sets: "maximal", "downscale", "upscale" and "minimal", which differ strongly in the number of training images and average number of pedestrians. We use 800 patches per image and, in order to compensate for differences in training set size, we optimized the tree depth. We note that our method performs only slightly worse than the state of the art method [9] where a hierarchical approach, with a classifier output from the first level serving as input to a second level, was used.

4. Conclusion and future directions

We have presented a simple and efficient method to estimate the density of objects in an image. Our method conceptually extends scalar regression random forests to output patches, a specific example of structured labels. We obtain results on par with state of the art algorithms, while conducting all experiments with the same architecture, simpler features, and minimal parameter tuning.

Interesting directions for future work include the learning from less precise, as well as from partial, annotations.

References

[1] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[2] A. Chan, Z.-S. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. *CVPR*, 2008.

[3] S. Cho, T. Chow, and C. Leung. A neural-based crowd estimation by hybrid global learning algorithm. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 29(4):535–541, 1999.

[4] A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning. *Microsoft Research technical report*, pages 1–151, 2011.

[5] L. Dong, V. Parameswaran, V. Ramesh, and I. Zoghlami. Fast crowd segmentation using shape indexing. *ICCV*, 2007.

[6] G. Flaccavento, V. Lempitsky, I. Pope, P. R. Barber, A. Zisserman, J. A. Noble, and B. Vojnovic. Learning to count cells: applications to lens-free imaging of large fields. In *Microscopic Image Analysis with Applications in Biology*, 2011.

[7] D. Kong, D. Gray, and H. Tao. A viewpoint invariant approach for crowd counting. *ICPR*, 2006.

[8] P. Kontschieder, S. Bulo, H. Bischof, and M. Pelillo. Structured class-labels in random forests for semantic image labelling. *ICCV*, 2011.

[9] V. Lempitsky and A. Zisserman. Learning to count objects in images. *NIPS*, 2010.

[10] A. Marana, S. Velastin, L. Costa, and R. Lotufo. Estimation of crowd density using image processing. In *Image Processing for Security Applications (Digest No.: 1997/074), IEE Colloquium on*, pages 11–1. IET, 1997.

[11] N. Meinshausen. Quantile regression forests. *The Journal of Machine Learning Research*, 7:983–999, 2006.

[12] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. *NIPS*, 2007.

[13] M. Rodriguez and E. N. Sup. Density-aware person detection and tracking in crowds. *Energy*, pages 2423–2430, 2011.

[14] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Crowd counting using multiple local features. In *Digital Image Computing Techniques and Applications*, 2009.

[15] B. Wu and R. Nevatia. Detection and segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses. *IJCV*, 82(2):185–204, 2008.