AN EFFICIENT NAIVE BAYES APPROACH TO CATEGORY-LEVEL OBJECT DETECTION

Kasim Terzić *J.M.H. du Buf*

Vision Lab (LARSyS), FCT, University of the Algarve, Gambelas Campus, 8000 Faro, Portugal

ABSTRACT

We present a fast Bayesian algorithm for category-level object detection in natural images. We modify the popular *Naive Bayes Nearest Neighbour* classification algorithm to make it suitable for evaluating multiple sub-regions in an image, and offer a fast, filtering-based alternative to the multi-scale sliding window approach. Our algorithm is example-based and requires no learning. Tests on standard datasets and robotic scenarios show competitive detection rates and real-time performance of our algorithm.

***Index Terms*—** Computer vision, Object detection, Real time systems, Robot vision, Nearest neighbour

1. INTRODUCTION

Object detection has always been one of the most important applications of Computer Vision. In recent years, research has moved from detection of particular objects [10] to category-level detection. Many standard datasets have appeared, from single-class detection problems [1] to complex datasets featuring dozens of classes [5]. A detailed survey of available algorithms is beyond the scope of this paper. Here we concentrate on a particular issue involved in many modern algorithms – the expensive search over positions and scales, especially when combined with non-parametric probability density estimation methods.

While there are some notable methods based on generating hypotheses [7, 9, 21, 24], most recent state-of-the-art methods in object detection use a sliding window approach for testing a vast number of possible object locations at multiple scales [4, 23, 25, 26]. Typically, powerful features are extracted from each region [4, 23], and these are classified using powerful classifiers such as SVMs [22] and boosting [23]. Most of these methods require a long learning stage, but once a model has been learned they are capable of fast classification, making it possible to test many hypotheses quickly.

In recent years, non-parametric naive Bayes methods based on nearest neighbours (NBNN) have shown competitive results in image and scene classification tasks [3, 12, 18, 19]. They estimate probability density during the recognition process, and classify using a MAP decision rule. They are attractive due to their simplicity, speed, absence of expensive learning, and Bayesian foundation. However, the complexity of such methods is concentrated in the classification step, making them a poor match for sliding window approaches.

In this paper, we present a reformulation of NBNN which transforms it from a pure classification algorithm into a multi-scale object detection method. It can be seen as a Naive Bayes interpretation of Hough voting methods like [9]. Our approach avoids sliding windows and thus incurs only a modest extra cost compared to the baseline NBNN algorithm. We evaluate the performance on several standardised benchmarks.

2. OBJECT DETECTION METHOD

Recent work has shown that an efficient data selection step can speed up object categorisation [18, 19]. We follow [18] and extract keypoints at several scales using the BIMP algorithm, and then extract SIFT descriptors at keypoint locations, with a size proportional to the keypoint scale. We then apply a modified version of the local NBNN algorithm of [12].

2.1. Local Naive Bayes Nearest Neighbour

We start by making the NBNN algorithm compatible with the sliding window approach. NBNN algorithms typically expand local descriptor vectors d_i (sampled at keypoint locations) with their scaled 2D image coordinates $\mathrm{x}(d_i)$ to obtain localised evidence feature vectors $e_i = [d_i, \alpha\mathrm{x}(d_i)]$, where α is a weight controlling the relative importance of a feature's position (see [3]). The implicit assumption is that the object is roughly centered in the image, so similar positions in the image correspond to similar positions on the object. Then log-likelihood is modelled by

$$\log P(e_1 \dots e_N|C) \approx \sum_{i=1}^{N} \|e_i - \mathrm{NN}_C(e_i)\|^2, \tag{1}$$

where $\mathrm{NN}_C(e_i)$ is the nearest neighbour of e_i among all descriptors obtained from images belonging to class C. There is no learning involved: all descriptors from all training images are kept. Nearest neighbours can be computed efficiently using fast linear approximate algorithms such as FLANN [15].

The above formulation is not yet useful for object detection because the location of each feature changes with the position of the sliding window, so all descriptors e_i change. We first modify Eqn 1 so that the descriptors d_i are separated from the keypoint locations $\mathrm{x}(d_i)$:

$$\log P(e_1 \dots e_N|C) \approx \sum_{i=1}^{N} \|d_i - \mathrm{NN}_C(d_i)\|^2 + \alpha \sum_{i=1}^{N} \|\mathrm{x}(d_i) - \mathrm{x}(\mathrm{NN}_C(d_i))\|^2 . \qquad (2)$$

Eqns 1 and 2 yield the same sums if the nearest neighbours of e_i and d_i are the same for all i. In practice, differences are small because e_i and d_i consist mostly of the same elements, and the contribution of $\mathrm{x}(d_i)$ to the total distance is small.

Folowing the approach of [12], we minimise the log-odds to obtain the winning class. This means that the actual number of keypoints used per class can vary:

$$c = \arg\min_C \sum_{i=1}^{N} \left(\|d_i - \mathrm{NN}_C(d_i)\|^2 - \mathrm{distb}(d_i)\right) + +\alpha \sum_{i=1}^{N} \|\mathrm{x}(d_i) - \mathrm{x}(\mathrm{NN}_C(d_i))\|^2 , \qquad (3)$$

where *distb* is the background distance estimate for descriptor d_i estimated using a nearby sample (see [12] for details).

We store original descriptors without 2D coordinates and for each one we keep the offset from the object centre, expressed as a function of the scale of the local keypoint, in a separate structure $\mathrm{s}(d_i)$. This allows for reusing the feature vectors and using a single FLANN structure for classifying any sub-region in an image. In principle, it is then possible to apply sliding windows for object detection.

2.2. Background Model

In order to reject regions which do not contain objects, we need to define a background class. For keeping our model general, we do not assume that the background can be "learned" from training images. We make two basic assumptions about the background: (i) background can look like anything, and (ii) background has no learnable structure:

$$P(d_i|\mathrm{bg}) = p_{\mathrm{bg}} \ ; \quad P(e_i|\mathrm{bg}) \approx p_{\mathrm{bg}}/N_p \ , \qquad (4)$$

where p_{bg} is a constant and N_p the number of pixels in the image. The reasoning behind this is that the inherent structure of objects will concentrate object hypotheses on a few spots, causing the conditional probability at those points to exceed the background probability. Constant p_{bg} can be approximated by counting features in training images, but we treat it as a parameter for controlling false positives.

2.3. Object Detection Using Filter Kernels

We now replace the standard sliding window approach by a voting scheme followed by two filtering operations (see

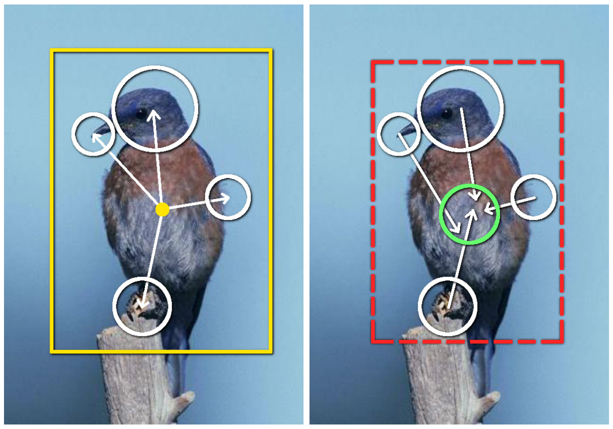

Fig. 1. In standard sliding window methods (left), a window is repeatedly positioned over image regions (yellow), and feature locations are expressed with respect to that window. In our approach (right), each feature "votes" for an object centre, and the votes are grouped by a filter kernel (green), thus replacing many classification decisions with a fast filtering operation. The final bounding box (red) is calculated by combining all features that voted for the same hypothesis.

Fig. 1). We set up two 2D matrices per class, with the same number of rows and columns as the image.[1] In the first step, we determine $\mathrm{NN}_C(d_i)$ for each class and each local keypoint descriptor d_i in the image using a single index [12].

If $\mathrm{x}(d_i)$ is the position of a local feature d_i and $\mathrm{s}(d_i)$ is its scaled offset from the object centre, then feature d_i "expects" an object of class C at 2D location $l_{i,C}$:

$$l_{i,C} = \mathrm{x}(d_i) - \mathrm{s}(\mathrm{NN}_C(d_i)) \ . \qquad (5)$$

Each of the local features d_i "votes" for the object centre by incrementing the values of the two matrices:

$$M_C^{\mathrm{v}}(l_{i,C}) := M_C^{\mathrm{v}}(l_{i,C}) + 1 \ ; \qquad (6)$$

$$M_C^{\mathrm{s}}(l_{i,C}) := M_C^{\mathrm{s}}(l_{i,C}) + \mathrm{dist}(d_i, \mathrm{NN}_C(d_i)) - \mathrm{distb}(d_i) \ . \qquad (7)$$

The first matrix M_C^{v} will contain the number of votes for each pixel as an object centre. The second matrix M_C^{s} will hold the sum of descriptor distances between the local features voting for that point and their nearest neighbours in the descriptor database. We now define two kernels with radius R:

$$K_{\mathrm{mask}}(x,y) = \begin{cases} 1 & \text{if } x^2+y^2 < R \\ 0 & \text{otherwise ;} \end{cases} \qquad (8)$$

$$K_{\mathrm{dist}}(x,y) = \begin{cases} x^2+y^2 & \text{if } x^2+y^2 < R \\ 0 & \text{otherwise .} \end{cases} \qquad (9)$$

These two kernels assume the L_2 distance metric. Corresponding L_1 kernels are obtained by dropping the square operator, which makes the kernels separable in x and y and results in a considerable improvement in speed. Finally, we

[1] Subsampling is possible for large images.

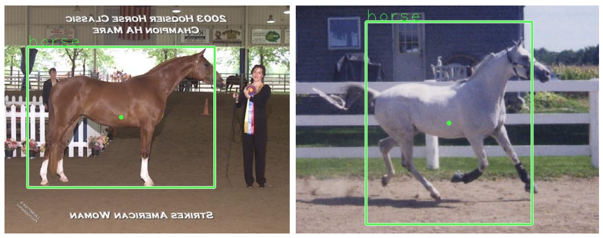

Fig. 2. Some detections on the INRIA Horses dataset [8].

create the object detection matrix for each class by convolving the two matrices with the kernels and summing them:

$$M_C = M_C^{\mathrm{s}} * K_{\mathrm{mask}} + \alpha M_C^{\mathrm{v}} * K_{\mathrm{dist}} \ . \tag{10}$$

α is the weight determining how much keypoint location contributes to the total sum. An object of class $c \in C$ is detected if the value of M_c at a local minimum is lower than the value of each $M_{c' \neq c}$ and lower than the background constant.

It is important to note that, for large values of R, Eqn 10 computes exactly the same sum as Eqn 3 for each element of M_C: the first convolution is a sum of descriptor distances, and the second convolution is a sum of weighted descriptor offsets. In fact, Eqn 10 is equivalent to evaluating Eqn 3 at each pixel location in the image, keeping the classification performance of NBNN while speeding up the sliding window search by orders of magnitude.

2.4. Localisation of Detected Objects

After detecting object centres, we estimate the boundaries of the object. We only allow one object to be present at any particular location in the image, so we group all points which voted for a particular hypothesis. Each point is associated with a training object and its bounding box. We set the width and height of the detected object to be the median width and height of the bounding boxes proposed by all collected points. Remaining hypotheses of all objects are sorted by total distance, and all hypotheses which overlap with a better hypothesis by more than 50% are eliminated (measured as intersection over union) [5].

2.5. Time Complexity

The complexity of the local NBNN classification algorithm in Eqn 2 is $O(cN_D \log(N_C N_T N_D))$ per image, where N_C is the number of classes, N_T the number of training images per class, N_D the average number of descriptors per image, and c the number of times a K-D tree should be traversed. The use of a sliding window expands this to $O(N_w c N_D \log(N_C N_T N_D))$, where N_w is the number of separate regions examined by the algorithm, typically in the hundreds. Ideally, we would perform the nearest neighbour lookup only once per descriptor.

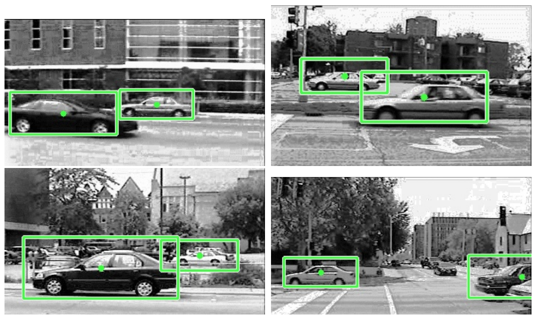

Fig. 3. Top row: some detections on the multi-scale UIUC dataset [1] are rejected by the evaluation software due to small imprecisions in the detected bounding box, but are accepted by the Pascal criterion. Bottom row: some detections of occluded cars were not annotated, resulting in false positives. Half of our false positives fall into this category.

The new algorithm reduces the complexity to $O(cN_D \log(N_C N_T N_D)) + O(\text{filtering})$. The filtering complexity is $O(N_C N_p \log N_p)$ for 2D filtering in the frequency domain (needed with L_2 distance), and $O(N_C N_p R)$ for separable filtering in the spatial domain (L_1 distance). N_p is the number of pixels in the image and R is the kernel radius. In practice, the filtering operation is very fast compared to the required nearest-neighbour lookups. Therefore, our new algorithm is capable of finding multiple objects belonging to multiple classes with only a very modest additional complexity compared to a simple local-NBNN classification.

3. EVALUATION

We evaluated our algorithm on two popular category-level detection benchmarks and a real-time robotic scenario.

3.1. UIUC Cars Dataset

The UIUC Cars dataset [1] consists of 550 positive training images, 500 negative training images, 170 single-scale testing images and 108 multi-scale testing images. Since all annotations have the same aspect ratio, we report a box with this ratio which maximises the overlap with our detected bounding box (see Fig. 3). We did not use the negative training images except for estimating *distb* in Eqn 3. Table 1 reports the F-score using $R = 25$ and compares it to other methods. In the single-scale test, our algorithm achieves state-of-the-art results, although it is competing against specialised classifiers. The official evaluation software assumes a sliding window of a fixed size, which is a poor match for our more general algorithm which estimates the bounding box directly from image data. Because of this, the official F-score on multi-scale images was only 46%. When using the more common Pascal overlap criterion, our method achieves 96%, i.e. a state-of-the-art result (see Table 1).

Table 1. F-score on the UIUC Cars dataset compared to some state-of-the-art approaches. The result marked with $*$ was obtained using the Pascal criterion.

detector	single scale	multi-scale
Agarwal *et al.* [2]	0.765	0.396
Leibe *et al.* [9]	0.975	
Our result	**0.983**	0.458 / **0.96***
Maji *et al.* [11]	0.985	
Mutch and Lowe [16]	0.999	0.906

Table 2. Recall on the horses dataset (averaged over five trials) at different *false positive per image* rates compared to state-of-the-art approaches.

detector	0.1 fppi	0.4 fppi	1 fppi
Ferrari *et al.* [6]	0.55	0.70	0.9
HoG+SVM [13]	0.66	0.82	0.91
BOSS [20]	0.65	0.84	0.92
Our result	**0.82**	**0.87**	**0.92**
Monroy & Ommer [14]	0.90	0.92	0.93

3.2. INRIA Horses

The INRIA Horses dataset [8] consists of 170 images containing horses and 170 negative images. Large variations in scale and pose make this a challenging category-level detection test. Following [6], we use 50 images for training and the rest for testing, with $R = 60$. Once again we do not make use of the background category other than for estimating *distb* and testing. Despite competing against much more complex discriminative shape models, our algorithm easily achieves state-of-the-art performance on this dataset, see Table 2.

3.3. Real-Time Detection in Robotics

Our algorithm is particularly suited to real-time robotics applications due to the absence of a slow learning step, the ability to expand object models on-line, and real-time performance. We tested the detection of ten household items often used in robotic scenarios on a video (Fig. 4). With only three examples per class at a single scale, our algorithm correctly detected the objects at all scales with no false positives.

3.4. Speed

We ran our algorithm on an i5 processor at 2.8 GHz and a GeForce 460 Ti GPU. On an image with 640$\times$480 pixels, our algorithm runs in real time: GPU-assisted keypoint extraction takes 24ms, SIFT feature extraction takes 50ms, and our object detection algorithm 110ms on average, for a total of 180ms. Using four cores of a modern processor, this gives 20 frames per second on average. Complete evaluation of the Cars and Horses datasets on the CPU took 1m and 2m, respectively. Most of that time was spent on feature extraction.

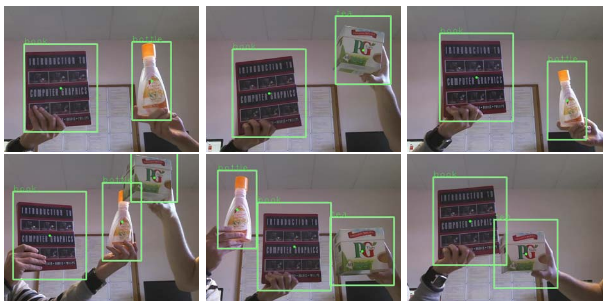

Fig. 4. Object detection in a real-time scenario. Only three views of each object were used as models and more can be added without an expensive re-learning step.

Since our algorithm exhibits logarithmic complexity with respect to the number of stored models and classes,[2] it scales well to many classes and models. It is important to stress that our algorithm does not employ classical machine learning – only efficient storage of many object views, making "learning" of new classes and object views almost instantaneous.

4. CONCLUSIONS

We presented a novel category-level object detection scheme based on the popular Naive Bayes Nearest Neighbour algorithm. We extended the original NBNN algorithm to obtain an efficient method for object localisation and detection. Our algorithm is interesting because it replaces the common sliding window approach with an efficient filtering operation, while retaining the excellent classification performance of NBNN. First results show good performance on a range of object classification and detection benchmarks while meeting the real-time constraint.

To the best of our knowledge, this is the fastest category-level object detection algorithm available, especially when learning time is also taken into account. It requires no classical learning, it can extend object models with minimal cost at any time (analogous to online learning), and it can detect dozens of classes in an image in real time. These properties make our algorithm attractive for scenarios in cognitive robotics, where performance is critical.

We are currently exploring biologically-motivated extensions of our algorithm in terms of dynamics and biological features, integrating our algorithm in a more complex scene representation system [17], and using our algorithm for landmark detection in a cognitive robot.

Acknowledgements This work was supported by the EU under the FP-7 grant ICT-2009.2.1-270247 *NeuralDynamics* and the FCT under the grant PEst-OE/EEI/LA0009/2011 .

[2]Filtering in Eqn 10 is linear with the number of classes, but this part is generally much faster than nearest-neighbour lookups.

5. REFERENCES

[1] S. Agarwal, A. Awan, and D. Roth. UIUC image database for car detection, 2002. http://l2r.cs.uiuc.edu/ cogcomp/Data/Car/ [Online; accessed 20-Jan-2014].

[2] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *T-PAMI*, 26(11):1475–1490, Nov 2004.

[3] O. Boiman, E. Shechtman, and M. Irani. In Defense of Nearest-Neighbor Based Image Classification. In *Proc. CVPR*, Anchorage, 2008.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.

[5] M. Everingham. The VOC 2006 database, 2006. http://pascallin.ecs.soton.ac.uk/challenges/VOC/databases.html [Online; accessed 20-Jan-2014].

[6] V. Ferrari, F. Jurie, and C. Schmid. From images to shape models for object detection. *IJCV*, 87(3):284–303, May 2010.

[7] C.H. Gu, J.J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR*, pages 1030–1037, 2009.

[8] F. Jurie and C. Schmid. Scale-invariant shape features for recognition of object categories. In *CVPR*, 2004.

[9] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[10] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.

[11] S. Maji, A. C. Berg, and J. Malik. Efficient classification for additive kernel SVMs. *T-PAMI*, 35(1), Jan 2013.

[12] S. McCann and D.G. Lowe. Local naive bayes nearest neighbor for image classification. In *CVPR*, pages 3650–3656, Providence, 2012.

[13] A. Monroy, A. Eigenstetter, and B. Ommer. Beyond straight lines - object detection using curvature. In *ICIP*, 2011.

[14] A. Monroy and B. Ommer. Beyond bounding-boxes: Learning object shape by model-driven grouping. In *ECCV*. Springer, 2012.

[15] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISSAPP*, pages 331–340. INSTICC Press, 2009.

[16] J. Mutch and D. G. Lowe. Object class recognition and localization using sparse features with limited receptive fields. *IJCV*, 80(1):45–57, October 2008.

[17] K. Terzić, D. Lobato, M. Saleiro, J. Martins, M. Farrajota, J.M.F. Rodrigues, and J.M.H. du Buf. Biological models for active vision: Towards a unified architecture. In *ICVS 2013, LNCS*, volume 7963, pages 113–122, St. Petersburg, Russia, Jul 2013. Springer.

[18] K. Terzić, J.M.F. Rodrigues, and J.M.H. du Buf. Fast cortical keypoints for real-time object recognition. In *ICIP*, pages 3372–3376, Melbourne, Sep 2013.

[19] R. Timofte, T. Tuytelaars, and L. J. Van Gool. Naive bayes image classification: Beyond nearest neighbors. In *ACCV (1)*, pages 689–703, 2012.

[20] A. Toshev, B. Taskar, and K. Daniilidis. Shape-based object detection via boundary structure segmentation. *IJCV*, 2012.

[21] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. Segmentation as selective search for object recognition. In *ICCV*, pages 1879–1886, 2011.

[22] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[23] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[24] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, 2013.

[25] A. Yuille Y. Chen, L. Zhu. Active mask hierarchies for object detection. In *ECCV*, 2010.

[26] Y. Yu, J. Zhang, Y. Huang, S. Zheng, W. Ren, K. Huang, and T. Tan. Object detection by context and boosted HOG-LBP. In *PASCAL VOC Workshop, ECCV*, Crete, Greece, 2010.