Boosting Bayesian MAP Classification

Paolo Piro
CNRS/University of Nice-Sophia Antipolis
piro@i3s.unice.fr

Richard Nock
CEREGMIA, University of Antilles-Guyane
rnock@martinique.univ-ag.fr

Frank Nielsen
Ecole Polytechnique, France / Sony CSL
nielsen@lix.polytechnique.fr

Michel Barlaud
CNRS/University of Nice-Sophia Antipolis
barlaud@i3s.unice.fr

Abstract

In this paper we redefine and generalize the classic k-nearest neighbors (k-NN) voting rule in a Bayesian maximum-a-posteriori (MAP) framework. Therefore, annotated examples are used for estimating pointwise class probabilities in the feature space, thus giving rise to a new instance-based classification rule. Namely, we propose to "boost" the classic k-NN rule by inducing a strong classifier from a combination of sparse training data, called "prototypes". In order to learn these prototypes, our MAPBOOST *algorithm globally minimizes a multiclass exponential risk defined over the training data, which depends on the class probabilities estimated at sample points themselves.*

We tested our method for image categorization on three benchmark databases. Experimental results show that MAPBOOST *significantly outperforms classic k-NN (up to 8%). Interestingly, due to the supervised selection of sparse prototypes and the multiclass classification framework, the accuracy improvement is obtained with a considerable computational cost reduction.*

1. Introduction

We address the task of image categorization, which aims at classifying images into a predefined set of categories. Several techniques have been proposed to solve this problem automatically, among which instance-based methods, like k-nearest neighbors (k-NN) classification, have shown very good performances [1]. In particular, much research effort has been devoted to improve the statistical and computational properties of the classic k-NN vote, which relies on labeled neighbors to predict the class of unlabeled data [11]. Such methods can be viewed as primers to improve the (continuous) estimation of the class membership probabilities. Moreover, a Bayesian reassessment of the problem has been recently proposed as a motivation for the formal transposition of *boosting* to k-NN classification [5].

We generalize the k-NN rule in a supervised Bayesian framework, where annotated data (*sample points*) are used for non-parametric *maximum-a-posteriori* (MAP) estimation [2]. Namely, our main contribution is redefining the classic voting rule as a strong classifier that linearly combines predictions from sample points in a boosting framework. For this purpose, our boosting algorithm minimizes a multiclass risk function over training data, thus redefining the UNN approach of [9] directly in a multiclass framework.

In the following sections, we first define the boosting problem for MAP classifiers and describe our leveraging algorithm (Sec. 2.1–2.2). Then, we provide the solution when using kernel density estimators (Sec. 2.4), thus enlightening the link to classic k-NN classification. Finally, we present and discuss some experimental results on categorization of natural images (Sec. 3).

2. Method

2.1 (Leveraged) MAP classification

We tackle the classification problem directly in a *multiclass* framework, *i.e.*, unlike [9], we do not reduce it to multiple two-class problems. We suppose given a training set $\mathcal{S}$ of m annotated examples $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, where $\boldsymbol{x}_i$ is the image descriptor and $\boldsymbol{y}_i$ the *class vector* that specifies the category membership. In particular, the sign of component y_{ic} gives the positive/negative membership of the example to class c ($c = 1, ..., C$). Inspired by the multiclass boosting analysis of [12], we constrain

1051-4651/10 $26.00 © 2010 IEEE
DOI 10.1109/ICPR.2010.167

the class vector to be *symmetric*, *i.e.*, $\sum_{c=1}^{C} y_{ic} = 0$ by setting: $y_{i\,\tilde{c}} = 1$, $y_{i\,c\neq\tilde{c}} = -\frac{1}{C-1}$, $\tilde{c}$ being the true image category.

The multiclass boosting algorithm we propose grounds on MAP classification, where, for a given $\boldsymbol{x}$, one selects the label that maximizes the *a-posteriori* probability, as it can be estimated from the annotated data:

$$P(c|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, c)}{\sum_{j=1}^{C} p(\boldsymbol{x}, j)} = \frac{\sum_{i:y_{ic}=1} \hat{f}_i(\boldsymbol{x})}{\sum_{i=1}^{m} \hat{f}_i(\boldsymbol{x})} \,. \quad (1)$$

In (1), $p(\boldsymbol{x}, c)$ is the joint probability density of observations and classes, and $\hat{f}_i(\boldsymbol{x})$ is a density estimation function evaluated at $\boldsymbol{x}$ for each example i. Note that only the examples belonging to class c contribute to the probability estimation, whereas the denominator serves simply as a normalizing factor.

MAP classification based on (1) can be viewed as a uniform voting (with real-valued votes $\hat{f}_i$) among sample points $\boldsymbol{x_i}$. We propose to replace the training dataset in (1) by a subset of T training examples (called *prototypes*), which are learned in a boosting framework, thus defining:

$$P_T^{\ell}(c|\boldsymbol{x}) = \frac{\sum_{t:y_t^{(c)}=1} \alpha_t \hat{f}_t(\boldsymbol{x})}{\sum_{t=1}^{T} \alpha_t \hat{f}_t(\boldsymbol{x})} \,. \quad (2)$$

Hence, each term $\hat{f}_t$ is weighted by a *leveraging* coefficient α_t, which measures the "confidence" of the corresponding prototype t as a sample point for MAP classification. We use the modified class probability estimation (2) for defining our *leveraged* MAP classifier in the framework of multiclass boosting [12]:

$$h_c^{\ell}(\boldsymbol{x}) = \frac{\sum_{t=1}^{T} \alpha_t y_{tc} \hat{f}_t(\boldsymbol{x})}{\sum_{t=1}^{T} \alpha_t \hat{f}_t(\boldsymbol{x})} \,. \quad (3)$$

Therefore, we consider each prototype t as a weak classifier that matches the symmetric constraint imposed by [12]: $\sum_{c=1}^{C} y_{tc} \hat{f}_t(\boldsymbol{x}) = 0$. Finally, the criterion for predicting the label of unknown samples is:

$$\hat{c} = \arg\max_c h_c^{\ell}(\boldsymbol{x}) = \arg\max \sum_{t=1}^{T} \alpha_t y_{tc} \hat{f}_t(\boldsymbol{x}) \,, \quad (4)$$

where we can remove the normalizing factor of (3).

In order to fit rule (3), we have to:

- learn prototypes and their weights α_t;
- estimate $\hat{f}_t(\boldsymbol{x})$ from those prototypes.

In the following sections we first present our learning algorithm, then describe some techniques we used for estimating $\hat{f}_t(\boldsymbol{x})$ directly from training data.

2.2 Boosting by multiclass risk minimization

In order to learn prototypes from training set $\mathcal{S}$, we exploit a common trick in boosting, *i.e.*, minimizing a particular upperbound of the risk functional on training data. In particular, we focus on the following multiclass exponential risk:

$$\varepsilon^{\exp}\left(\boldsymbol{h}^{\ell}, \mathcal{S}\right) \doteq \frac{1}{m} \sum_{i=1}^{m} \exp\left\{-\rho(\boldsymbol{h}^{\ell}, i)\right\} \,, \quad (5)$$

where

$$\rho(\boldsymbol{h}^{\ell}, i) \doteq \frac{1}{C} \sum_{c=1}^{C} y_{ic} h_c^{\ell}(\boldsymbol{x}_i) \quad (6)$$

is the multiclass *edge* of classifier $\boldsymbol{h}^{\ell}$ on example $\boldsymbol{x}_i$. This edge averages over all classes quantity $y_{ic} h_c^{\ell}(\boldsymbol{x_i})$, which is positive iff the category membership predicted by the classifier agrees with the true membership of the example. (5) is an upper bound of the true *empirical risk*, thus acting as a convenient primer for the optimization problem [6].

Like common boosting algorithms, our method consists of an iterative minimization procedure. Namely, on each iteration t of the algorithm a new prototype j is selected, thus updating the (unnormalized) strong classifier (3) as follows:

$$h_c^{(t)}(\boldsymbol{x}_i) = h_c^{(t-1)}(\boldsymbol{x}_i) + \alpha_t y_{jc} \hat{f}_j(\boldsymbol{x}_i) \,. \quad (7)$$

Then, computing the leverage coefficient α_t associated to that prototype requires plugging (7) into the risk definition (5, 6), thus turning to the following convex optimization problem:

$$\arg\min_{\alpha_t} \sum_{i=1}^{m} w_i \cdot \exp\left\{-\alpha_t \hat{\mathrm{r}}_{ij}\right\} \,. \quad (8)$$

In (8), $w_i > 0$ are the weights defined over training data, which are repeatedly updated as they depend on the first $t-1$ weak classifiers:

$$w_i = \exp\left\{-\frac{1}{C} \sum_{c=1}^{C} y_{ic} h_c^{(t-1)}(\boldsymbol{x_i})\right\} \,, \quad (9)$$

whereas $\hat{\boldsymbol{R}} = [\hat{\mathrm{r}}_{ij}]_{m\times m}$ is the so-called *edge matrix*:

$$\hat{\mathrm{r}}_{ij} = \hat{f}_j(\boldsymbol{x}_i) \left(\frac{1}{C} \sum_{c=1}^{C} y_{ic} y_{jc}\right) \,. \quad (10)$$

This edge matrix is constant along iterations, as it only depends on the training data and the chosen density estimator. Moreover, $\hat{\mathrm{r}}_{ij}$ is positive iff labels of the two examples agree, whereas its absolute value depends on

the real-valued estimation from prototype j at $\boldsymbol{x_i}$. Due to this definition, the multiclass edge (6) on example i reads as a linear combination of the edges between i and the training data (indexed by j), with real coefficients α_j:

$$\rho(\boldsymbol{h}^\ell, i) = \sum_{j=1}^{m} \alpha_j \hat{\mathrm{r}}_{ij} \; . \tag{11}$$

Finally, fitting α_t in (7) amounts to taking the derivative of (8) and solving the following equation:

$$\sum_{i=1}^{m} w_i \hat{\mathrm{r}}_{ij} \exp\left\{-\alpha_t \hat{\mathrm{r}}_{ij}\right\} = 0 \; . \tag{12}$$

The convexity of optimization program (8) guarantees the uniqueness of the solution for (12). However, this solution can be written in closed form only in the simple case of uniform kernel, $\hat{\mathrm{r}}_{ij}$ being constant over all i's:

$$\alpha_t \leftarrow \frac{(C-1)^2}{C} \log\left(\frac{(C-1)w_j^+}{w_j^-}\right) , \tag{13}$$

with:

$$w_j^+ = \sum_{i:\ \hat{\mathrm{r}}_{ij}>0} w_i, \quad w_j^- = \sum_{i:\ \hat{\mathrm{r}}_{ij}<0} w_i \; . \tag{14}$$

Otherwise, in the general case, one has to compute α_t numerically, and for this purpose we propose to use Newton's recursive formula:

$$\alpha_t^{(k+1)} = \alpha_t^{(k)} + \frac{\sum_{i=1}^m w_i \hat{\mathrm{r}}_{ij} \exp\left\{-\alpha_t^{(k)} \hat{\mathrm{r}}_{ij}\right\}}{\sum_{i=1}^m w_i \hat{\mathrm{r}}_{ij}^2 \exp\left\{-\alpha_t^{(k)} \hat{\mathrm{r}}_{ij}\right\}} \; . \tag{15}$$

2.3 Algorithm properties and convergence

Alg. 1 is pseudocode of our MAPBOOST algorithm for learning prototypes and their leveraging coefficients by minimizing (5). Our algorithm repeatedly updates weights w_i in order to progressively decrease the exponential risk function (5) over the training data. In particular, on each boosting iteration t a *weak index chooser* oracle $\textsc{Wic}(\{1,2,...,m\},t)$ determines index $j \in \{1,2,...,m\}$ of the example to leverage (step I.0), which is then included in the prototype set (step I.3). Notice that various choices are possible for this oracle. The simplest is computing Eq. (12) for all the training examples, then picking j that maximizes α_t. Indeed, α_t in Eq. (12) can be viewed as a local measure of the class density, which is as better as α_t gets large. (See [6], Lemma 4.)

Using known arguments of the boosting theory [6], we proved the convergence of our boosting MAP algorithm to the minimum of the surrogate risk, along with a convergence rate, which is based on the following *weak index assumption* (**WIA**):

(**WIA**) let $p_j \doteq w_j^+/(w_j^+ + w_j^-)$. There exist some $\gamma > 0$ and $\eta > 0$ such that the following two inequalities holds for index j returned by $\textsc{Wic}(\{1,2,...,m\},t)$:

$$|p_j - 1/C| \;\geq\; \gamma \; , \tag{16}$$

$$(w_j^+ + w_j^-)/||\boldsymbol{w}||_1 \;\geq\; \eta \; . \tag{17}$$

We summarize this fundamental convergence property in the following theorem:

Theorem 1 *If the **WIA** holds for $\tau \leq T$ steps, then* MAPBOOST *converges with τ to $\boldsymbol{h}^\ell$ realizing the **global** minimum of the surrogate risk (5), and $\varepsilon^{0/1}(\boldsymbol{h}^\ell, \mathcal{S}) \leq \exp(-\frac{C}{C-1}\eta\gamma^2\tau)$.*

Because we consider examples themselves as weak classifiers, inequality (16), which is the usual weak learning assumption, is not enough for guaranteeing convergence. Indeed, we also require a *weak coverage assumption* (17) to be matched, because insufficient *coverage* of the reciprocal neighbors could easily wipe out the surrogate risk reduction due to a large γ in (16).

Algorithm 1: MAPBOOST ($\mathcal{S}$)

Input: $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\ ,\ i = 1,2,...,m\ ,\ \boldsymbol{y}_i \in \{-\frac{1}{C-1}, 1\}^C\}$

Let:

$$\hat{\mathrm{r}}_{ij} \doteq \hat{f}_j(\boldsymbol{x}_i)\left(\frac{1}{C}\sum_{c=1}^{C} y_{ic}y_{jc}\right)$$

Let $\quad w_i \leftarrow 1/m, \quad \forall i = 1,2,...,m$

for $t = 1,2,...,T$ **do**

[I.1] Weak index chooser oracle:
Let $j \leftarrow \textsc{Wic}(\{1,2,...,m\},t)$;
Select prototype $(\boldsymbol{x}_t, \boldsymbol{y}_t) \leftarrow (\boldsymbol{x}_j, \boldsymbol{y}_j)$

[I.2] Compute α_t solution of:

$$\sum_{i=1}^{m} w_i \hat{\mathrm{r}}_{ij} \exp\left\{-\alpha_t \hat{\mathrm{r}}_{ij}\right\} = 0 \; ; \tag{18}$$

[I.3] Let

$$w_i \leftarrow w_i \exp(-\alpha_t \hat{\mathrm{r}}_{ij}), \quad \forall i = 1,2,...,m \; ; \tag{19}$$

Output:

$$h_c^\ell(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t y_{tc} \hat{f}_t(\boldsymbol{x}) \; .$$

2.4 Kernel density estimation

In order to compute $\hat{f}_j(\boldsymbol{x}_i)$ in (10), we propose to use a non-parametric density estimator, which relies on

kernel $K(\boldsymbol{x}, \boldsymbol{x}_i)$ [10]:

$$\hat{f}(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m}\hat{f}_i(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{h(\cdot)^d}\,K\left(\frac{||\boldsymbol{x}-\boldsymbol{x}_i||}{h(\cdot)}\right). \tag{20}$$

($h(\cdot)$ is the scale parameter of the kernel.) Therefore, plugging (20) into (10) gives:

$$\hat{\mathrm{r}}_{ij} = \left(\frac{1}{C}\sum_{c=1}^{C} y_{ic}y_{jc}\right)\frac{K\left(\frac{||\boldsymbol{x}-\boldsymbol{x}_i||}{h(\cdot)}\right)}{\sum_{i=1}^{m}K\left(\frac{||\boldsymbol{x}-\boldsymbol{x}_i||}{h(\cdot)}\right)}. \tag{21}$$

Different kinds of estimators can be defined using (20), depending on the way $h(\cdot)$ is chosen: (I) *balloon estimator*, where $h = h(\boldsymbol{x})$, *i.e.*, a single variable-size kernel centered at the estimation point; (II) *sample-point estimator*, where $h = h(\boldsymbol{x}_i)$, *i.e.*, a mixture of identical but individually scaled kernels that are centered at each observation $\boldsymbol{x}_i$.

We use the sample point estimator for computing the training edge matrix (10), and the balloon estimator for the classification rule (3). In the following, we concentrate on rectangular and Gaussian kernels, and set $h(\boldsymbol{x}_i) = \rho_k(\boldsymbol{x}_i)$, $\rho_k(\boldsymbol{x}_i)$ being the Euclidean distance from $\boldsymbol{x}_i$ to its k-th nearest neighbor in the dataset. (We also truncated the Gaussian kernel at $h(\boldsymbol{x}_i)$.) As a result of this setting, estimator (20) with rectangular kernel amounts to a uniform voting rule based on k-nearest neighbors. This allows us to consider Alg. 1 as a multiclass generalization of the two-class UNN boosting algorithm described in [9]. In particular, allowing a different kernel from the rectangular one can be viewed as generalizing the k-NN classification to a weighted voting rule, where geometric information between neighbors data in the feature space is taken into account as well.

3 Experiments

We evaluated our MAPBOOST algorithm for image categorization. We tested our method with both rectangular and Gaussian kernels. In particular, we compared two different versions of the Gaussian kernel estimator, the one relying on a variable-size kernel (with parameter scale h equal to the k-NN distance), the other using a fixed-size kernel, *i.e.*, h being independent on the local sample density. In the following, we refer to these approaches as "adaptive Gaussian" and "fixed Gaussian", respectively. In both cases, we truncated kernels at the k-th nearest neighbor. We also tested MAPBOOST in a one-versus-all classification framework, where the learning problem is decomposed in multiple binary problems (as many as the number of classes), *i.e.*, one class is trained at a time against all the others. Finally, we compared our method with both uniform and weighted k-NN classification. For this latter method, we tested the weighting rule proposed by Philbin et al. [8], where the votes are weighted by a Gaussian function of the distance between k-NN and the query point.

We validated our algorithm on three well-known image categorization databases: *8-cat* [7], *13-cat* [3] and *15-cat* [4], which contain, respectively: 2,688 images (in 8 categories), 3,759 images (in 13 categories) and 4,385 images (in 15 categories). We splitted each database in two subsets, one for training (containing 2,000 randomly selected images), the other for test. Images were represented using Gist, a state-of-the-art global image descriptor which has been shown to be very discriminant for classifying natural scenes [7]. Namely, we extracted 320 features from an image to form a Gist descriptor, then reduced the descriptor dimension down to 128 by means of PCA.

Fig. 1(a) shows performances of our boosting algorithm with (both adaptive and fixed) Gaussian and rectangular kernel, as well as those of one-vs-all MAPBOOST and the two k-NN methods. Classification performances are evaluated in terms of the mean Average Precision (mAP), *i.e.*, the average of the classification rates per category, and are reported as a function of the number of boosting rounds T, *i.e.*, the number of selected prototypes (ranging between 10% and 50% of the overall training set size). So as for k-NN classification, we selected a random sample of the training set and averaged results over a number of random sampling realizations. All the results we present were obtained with $k = 11$.

Three main empirical observations can be pointed out, which arise from these experimental results.

- Our method significantly outperforms both uniform and weighted k-NN voting, even when using a sparse prototype dataset (*e.g.*, in Fig. 1(a) remark 6% gap between MAPBOOST with adaptive Gaussian kernel and weighted k-NN at $T = 200$.)
- The Gaussian adaptive kernel provides the best performances overall (81% mAP on 8-cat, 67% on 13-cat, 64% on 15-cat), and generally outperforms both the fixed-size one and the adaptive rectangular one. This observation suggests that improving the accuracy of local density estimation, as enabled by a non-uniform adaptive kernel estimator, may improve the classification accuracy as well.
- The multiclass version of MAPBOOST outperforms the one-versus-all strategy, while dramatically reducing the computational cost, as it avoids

running the same algorithm C times independently on as many binary problems.

To summarize, experimental results enlight the ability of MAPBOOST to efficiently reject "noisy" examples, which are generally responsible for the failure of classic voting methods, thus improving both classification accuracy and computational speed at query time.

4 Conclusion

In this paper we have proposed MAPBOOST, a new multiclass boosting algorithm for Bayesian MAP classification. Our method induces a strong classifier by combining examples themselves as weak classifiers, thus generalizing classic k-NN classification. MAPBOOST significantly outperforms k-NN voting for image categorization, while considerably reducing the computational cost at classification time, thanks to the sparsity of the prototype set selected from training data.

References

[1] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *CVPR*, 2008.

[2] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. 2000.

[3] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *CVPR*, 2005.

[4] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[5] J.-M. Marin, C.-P. Robert, and D.-M. Titterington. A Bayesian reassessment of nearest-neighbor classification. *J. of the Am. Stat. Assoc.*, 2009.

[6] R. Nock and F. Nielsen. Bregman divergences and surrogates for learning. *IEEE PAMI*, 31:2048–2059, 2009.

[7] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, May 2001.

[8] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008.

[9] P. Piro, R.Nock, F.Nielsen, and M.Barlaud. Boosting k-NN for categorization of natural scenes. *ArXiv:1001.1221*, 2009.

[10] G. R. Terrell and D. W. Scott. Variable kernel density estimation. *Annals of Statistics*, 1992.

[11] H. Zhang, A. C. Berg, M. Maire, and J. Malik. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *CVPR*, 2006.

[12] J. Zhu, H. Zou, S. Rosset, and T. Hastie. Multi-class AdaBoost. *Statistics and Its Interface*, 2(3):349–360, 2009.

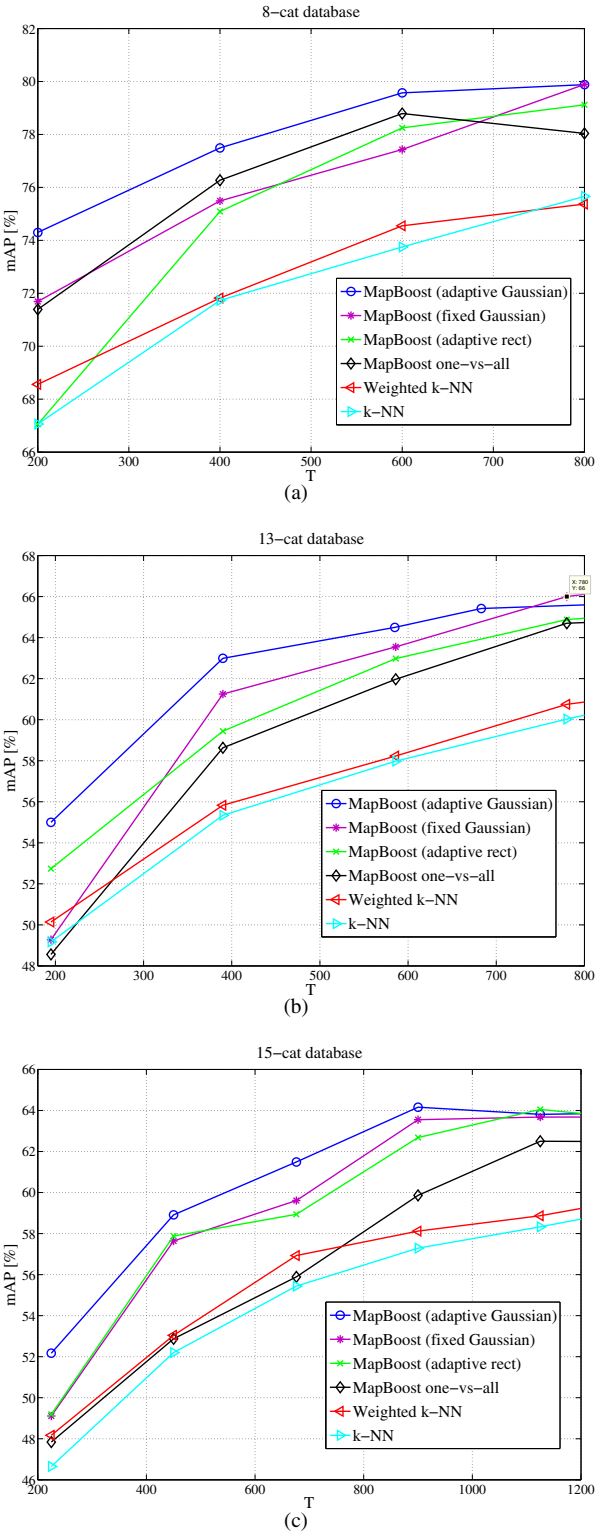

Figure 1. Categorization performances in terms of mAP (average of diagonal entries in the confusion matrix) as a function of the prototype set size T.