Towards Good Practice in Large-Scale Learning for Image Classification

Florent Perronnin[a], Zeynep Akata[a,b], Zaid Harchaoui[b] and Cordelia Schmid[b]
[a] TVPA team, XRCE, France [b] LEAR team*, INRIA, France

Abstract

We propose a benchmark of several objective functions for large-scale image classification: we compare the one-vs-rest, multiclass, ranking and weighted average ranking SVMs. Using stochastic gradient descent optimization, we can scale the learning to millions of images and thousands of classes. Our experimental evaluation shows that ranking based algorithms do not outperform a one-vs-rest strategy and that the gap between the different algorithms reduces in case of high-dimensional data. We also show that for one-vs-rest, learning through cross-validation the optimal degree of imbalance between the positive and the negative samples can have a significant impact. Furthermore, early stopping can be used as an effective regularization strategy when training with stochastic gradient algorithms. Following these "good practices", we were able to improve the state-of-the-art on a large subset of 10K classes and 9M of images of ImageNet from 16.7% accuracy to 19.1%.

1. Introduction

Large-scale image classification has recently received significant interest [11, 19, 28, 29]. This goes in hand with large-scale datasets being available. For instance, ImageNet (www.image-net.org) consists of more than 14M images labeled with almost 22K concepts [12].

Current state-of-the-art methods for large-scale image classification [19, 29] use high-dimensional image descriptors in combination with linear classifiers. The use of linear classifiers is motivated by their computational efficiency — a requirement when dealing with a large number of classes and images. High dimensional descriptors allow to separate the data with a linear classifier, i.e., they perform the feature mapping explicitly and avoid using non-linear kernels. As a rule of thumb, linear classifiers with high dimensional descriptors perform similarly to low dimensional bag-of-visual-words (BOV) with non-linear classifiers [9]. In the literature several high dimensional descriptors exist, *e.g.*, the Fisher vector [24, 26], local coordinate coding [37] and supervector coding [42]. A detailed comparison of these high-dimensional descriptors was performed in [9]; it showed that the Fisher vector representation outperforms the others. We use this image representation in the following and focus on large-scale learning of classifiers.

Most image classification approaches have adopted the simple strategy to train an *independent one-vs-rest binary* classifier for each class. A benefit of this approach is that the classifiers can be learned in parallel. This is a strong argument in our large-scale scenario since it can significantly reduce the training time. As an example, the two top systems at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2010 [5] used such an approach [19, 29].

A popular approach in the computer vision community is to view image classification as a ranking problem: given an image, the goal is to rank the labels according to their relevance. Performance measures such as the top-k accuracy which is used to report results on standard benchmarks, *e.g.* in ILSVRC, reflect this goal. While the one-vs-rest strategy is computationally efficient and yields competitive results in practice, it is clearly suboptimal with respect to a strategy optimizing directly a ranking loss [6, 10, 16, 35, 40, 41].

In this paper, we examine if these ranking approaches scale well to large datasets and if they improve the performance. We compare the one-vs-rest binary SVM, the multiclass SVM of Crammer and Singer [10] which optimizes top-1 accuracy, the ranking SVM of Joachims [16] which optimizes the rank of the labels as well the recent weighted approximate ranking of Weston *et al*. [38] which optimizes the top of the ranking list. The datasets we consider are large-scale in the number of classes (up to 10K), images (up to 9M) and feature dimensions (up to 130K). For efficiency reasons we train our linear classifiers using Stochastic Gradient Descent (SGD) algorithms [8, 18] with the primal formulation of the objective functions as in [30] for binary SVMs or in [23] for structured SVMs. By using the exact same optimization framework, we truly focus on the merits of the different objective functions, not on the merits of the particular optimization techniques.

Our experimental evaluation shows that ranking based algorithms do not outperform a one-vs-rest strategy and that

*The LEAR team is partially funded by the European integrated project AXES.

RECOMMENDATIONS FOR LARGE-SCALE IMAGE CLASSIFICATION

1. *Stochastic training*: learning with stochastic gradient descent is well-suited for large-scale datasets
2. *Class imbalance*: optimizing the imbalance parameter in one-vs-rest strategy is a must for competitive performance
3. *Early stopping*: regularizing through early stopping results in fast training and good generalization performance
4. *Step-size*: a small-enough fixed step-size w.r.t learning rate is often sufficient for state-of-the-art performance
5. *One-vs-rest*: one-vs-rest strategy is a flexible option for large-scale image classification
6. *Capacity saturation*: for sufficiently large feature representation all strategies lead to similar performance

the gap between the different algorithms reduces in case of high-dimensional data. Our experiments also show that, in the case of the one-vs-rest strategy, learning through cross-validation the optimal degree of imbalance between the positive and the negative examples can have a significant impact. Furthermore, early stopping can be used as an effective regularization strategy for fast training with SGD. Following these "good practices", we were able to improve the state-of-the-art on a a large subset of 10K classes and 9M of images of ImageNet [12] from 16.7% accuracy to 19.1%. We summarize our findings in the "recommendation box" at the top of this page.

The next section reviews related work and Section 3 presents the different objective functions used in our evaluation. Section 4 describes the SGD-based optimization. Experimental results for two large subsets of the ImageNet dataset, ILSVRC 2010 and ImageNet10K, are presented in Section 5. The code used for our experimental evaluation is available at `http://lear.inrialpes.fr/software`.

2. Related Work

In the following we present related work for large-scale image classification. Some approaches use simple classifiers such as nearest neighbor (NN) [32]. While exact NN can provide a competitive accuracy when compared to SVMs [11, 38], it is difficult to scale to large datasets. On the other hand, approximate nearest neighbor (ANN) can perform poorly on high-dimensional image descriptors (significantly worse than one-vs-rest SVMs) while still being much more computationally intensive [38].

For these reasons, the vast majority of the literature on large-scale image classification has employed large-margin classifiers. A fair amount of work has been devoted to scaling the learning algorithms to large datasets. An explicit mapping of the image descriptors to efficiently deal with non-linear kernels was proposed in [21, 25, 36]. Torresani *et al*. [33] used compact binary attribute descriptors to handle a large number of images. Sánchez and Perronnin [29] argued that high-dimensional image descriptors are necessary to obtain state-of-the-art results in large-scale classification and proposed to combine image descriptor compression with learning based on stochastic gradient descent. We underline that in most previous works tackling large-scale datasets, the objective function which is optimized is always the same: one binary SVM is learned per class in a one-vs-rest fashion [11, 19, 28, 29]. One-vs-rest strategies offer several advantages —see, *e.g.*, [27], for a defense of such strategies.

One noticeable exception to this rule is the large-scale ranking algorithm of Weston *et al*. [38] inspired by [35]. In their work, Weston *et al*. report on a subset of 15K ImageNet categories a significant increase of accuracy when optimizing a ranking objective function compared to one one-vs-rest: from 2.27% top-1 accuracy to 4.25%. We included this ranking algorithm in our benchmark.

There has also been a significant amount of work on reducing the computational cost of large-scale classification. For instance, Weston *et al*. proposed to learn jointly the classifier as well as a dimensionality reduction of the features [38]. To make the complexity sublinear in the number of classes, various approaches have been proposed which employ tree structures [4, 14, 22]. These approaches are outside the scope of our paper.

3. Objective Functions

Let us first introduce a set of notations. Let $S = \{(\mathbf{x}_i, y_i), i = 1 \dots N\}$ be the training set where $\mathbf{x}_i \in \mathcal{X}$ is an image descriptor, $y_i \in \mathcal{Y}$ is the associated label and $\mathcal{Y}$ is the set of possible labels. We shall always take $\mathcal{X} = \mathbb{R}^D$.

A learning strategy corresponds to an empirical risk minimization with a regularization penalty as follows

$$\underset{\mathbf{W}}{\text{Minimize}} \ \frac{\lambda}{2}\Omega(\mathbf{W}) + L(S;\mathbf{W}) \ , \qquad (1)$$

where $\mathbf{W}$ is the weight matrix stacking the weight vectors corresponding to each subproblem. The objective decomposes into the empirical risk $L(S;\mathbf{W}) := \frac{1}{N}\sum_{i=1}^{N} L(\mathbf{x}_i, y_i; \mathbf{W})$, with $L(\mathbf{x}_i, y_i; \mathbf{W})$ a surrogate loss of the labeled example (x_i, y_i), and the regularization penalty $\Omega(\mathbf{W}) := \sum_{c=1}^{C} \|\mathbf{w}_c\|^2$. The parameter $\lambda \geq 0$ controls the trade-off between the empirical risk and the regularization penalty.

We first briefly review the classical binary SVM. We then proceed with the multiclass, ranking and weighted approximate ranking SVMs. We finally discuss the issue of data re-weighting.

Binary One-Vs-Rest SVM (OVR) . In the case of the one-vs-rest SVM, we assume that we have only two classes and $\mathcal{Y} = \{-1, +1\}$. Let $\mathbb{1}(u) = 1$ if u is true and 0 otherwise. The 0-1 loss $\mathbb{1}(y_i \mathbf{w}^T x_i < 0)$ is upper-bounded by $L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) = \max\{0, 1 - y_i \mathbf{w}^T \mathbf{x}_i\}$. If we have more than two classes, then one transforms the C-class problem into C binary problems and trains independently C one-vs-rest classifiers.

3.1. Beyond binary

From now on, we treat the classes jointly and $\mathcal{Y} = \{1, \ldots, C\}$. Let $\{\mathbf{w}_c, c = 1 \ldots C\}$ denote the C classifiers corresponding to each of the C classes. In this case, $\mathbf{W}$ is a $C \times D$ dimensional vector obtained by concatenating the different w_c's. We denote by $\Delta(y, \bar{y})$ the loss incurred for assigning label $\bar{y}$ while the correct label was y. In this work, we focus on the 0/1 loss, *i.e.* $\Delta(y, \bar{y}) = 0$ if $y = \bar{y}$ and 1 otherwise. In what follows, we assume that we have one label per image to simplify the presentation.

Multiclass SVM (MUL). There exist several flavors of the multiclass SVM including the Weston and Watkins [39] and the Crammer and Singer [10] formulations (see [31] for a comprehensive review). Both variants propose a convex surrogate loss to $\Delta(y_i, \hat{y}_i)$ with $\hat{y}_i = \arg\max_y \mathbf{w}_y^T x_i$, *i.e.* the loss incurred by taking the highest score as the predicted label. We choose the Crammer and Singer formulation, corresponding to $L_{\text{MUL}}(\mathbf{x}_i, y_i; \mathbf{w}) = \max_y \left\{\Delta(y_i, y) + \mathbf{w}_y^T x_i\right\} - \mathbf{w}_{y_i}^T x_i$, which provides a tighter bound on the misclassification error [34]. Note that this can be viewed as a particular case of the structured SVM [34].

Ranking SVM (RNK). Joachims [16] considers the problem of ordering pairs of documents. Given a sample $(\mathbf{x}_i, y_i)$ and a label $y \neq y_i$, the goal is to enforce $\mathbf{w}_{y_i} \mathbf{x}_i > \mathbf{w}_y^T \mathbf{x}_i$. The rank of label y for sample $\mathbf{x}$ can be written as $r(\mathbf{x}, y) = \sum_{c=1}^{C} \mathbb{1}(\mathbf{w}_c^T \mathbf{x} \geq \mathbf{w}_y^T \mathbf{x})$. Given the triplet $(\mathbf{x}_i, y_i, y)$, $\mathbb{1}(\mathbf{w}_c^T \mathbf{x} \geq \mathbf{w}_y^T \mathbf{x})$ is upper-bounded by:

$$L_{\text{tri}}(\mathbf{x}_i, y_i, y; \mathbf{w}) = \max\{0, \Delta(y_i, y) - \mathbf{w}_{y_i}^T \mathbf{x}_i + \mathbf{w}_y^T \mathbf{x}_i\}.$$

Therefore, the overall loss of $(\mathbf{x}_i, y_i)$ writes as:

$$L_{\text{RNK}}(\mathbf{x}_i, y_i; \mathbf{w}) = \sum_{y=1}^{C} \max\{0, \Delta(y_i, y) - (\mathbf{w}_{y_i} - \mathbf{w}_y)^T \mathbf{x}_i\}.$$

Weighted Approximate Ranking SVM (WAR). An issue with the previous objective function is that the loss is the same when going from rank 99 to 100 or from rank 1 to rank 2. However, in most practical applications, one is interested in the top of the ranked list. Usunier *et al.* [35] therefore proposed to minimize a function of the rank which gives more weight to the top of the list. Let $\alpha_1 \geq \alpha_2 \geq \ldots \alpha_C \geq 0$ be a set of C coefficients. For sample $(\mathbf{x}_i, y_i)$ the loss is $\ell_{r(\mathbf{x}_i, y_i)}$ with ℓ defined as $\ell_k = \sum_{j=1}^{k} \alpha_j$. The penalty incurred by going from rank k to $k+1$ is α_k. Hence, a decreasing sequence $\{\alpha_j\}_{j \geq 1}$ implies that a mistake on the rank when the true rank is at the top of the list incurs a higher loss than a mistake on the rank when the true rank is lower in the list.

While [35] proposes an upper-bound on the loss, Weston *et al.* [38] propose an approximation. We follow [38] which is more amenable to large-scale optimization and write:

$$L_{\text{WAR}}(\mathbf{x}_i, y_i; \mathbf{w}) = \sum_{y=1}^{C} \ell_{r_\Delta(\mathbf{x}_i, y_i)} \frac{L_{\text{tri}}(\mathbf{x}_i, y_i, y; \mathbf{w})}{r_\Delta(\mathbf{x}_i, y_i)}, \quad (2)$$

where $r_\Delta(\mathbf{x}, y) = \sum_{c=1}^{C} \mathbb{1}(\mathbf{w}_c^T x + \Delta(y, c) \geq \mathbf{w}_y^T x)$ is a regularized rank. Following [38], we choose $\alpha_j = 1/j$. As opposed to other works [40, 41], this does not optimize directly standard information retrieval measures such as Average Precision (AP). However, it mimics their behavior by putting emphasis on the top of the list, works well in practice and is highly scalable.

3.2. Data reweighting

When the training set is unbalanced, *i.e.* when some classes are significantly more populated than others, it can be beneficial to reweight the data. This unbalance can be extreme in the one-vs-rest case when one has to deal with a large number of classes C as the unbalance between the positive class and the negative class is (on average) $C - 1$. In the binary case, the reweighting can be performed by introducing a parameter ρ and the empirical risk then writes as:

$$\frac{\rho}{N_+} \sum_{i \in I_+} L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) + \frac{1 - \rho}{N_-} \sum_{i \in I_-} L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) \quad (3)$$

where I_+ (resp. I_-) is the set of indices of the positive (resp. negative) samples and N_+ (resp. N_-) is the cardinality of this set. Note that $\rho = 1/2$ corresponds to the natural rebalancing of the data, *i.e.* in such a case one gives as much weight to positives and negatives.

When training all classes simultaneously, introducing one parameter per class is computationally intractable. It would require to cross-validate C parameters *jointly*, including the regularization parameter. In such a case, the natural re-balancing appears to be the most natural choice.

In the multiclass case, the empirical loss becomes:

$$\frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i\in I_c} L_{\text{MUL}}(\mathbf{x}_i, y_i; \mathbf{w}) \tag{4}$$

where $I_c = \{i : y_i = c\}$ and N_c is the cardinality of this set. One can perform a similar rebalancing in the case of L_{RNK} and L_{WAR}.

4. Optimization Algorithms

We now consider the optimization of the objective functions. To handle large datasets, we employ Stochastic Gradient Descent (SGD) [8] which has recently gained popularity in image classification [19, 25, 26, 28, 29, 38]. Since we deal with linear classifiers, we can perform the optimization directly in the primal. In the following, we describe the optimization algorithms for various objective functions and give implementation details. Finally, a brief comparison with batch solvers is reported.

4.1. Stochastic training

Training with stochastic gradient descent (SGD) consists at each step in choosing a sample at random and updating the parameters $\mathbf{w}$ using a sample-wise estimate of the regularized risk. In the case of R_{OVR} and R_{MUL}, the sample is simply a pair $(\mathbf{x}_i, y_i)$ while in the case of R_{RNK} and R_{WAR} it consists in a triplet $(\mathbf{x}_i, y_i, \bar{y})$ where $\bar{y} \neq y_i$. Let z_t denote the sample drawn at step t (whether it is a pair or a triplet) and let $R(z_t; \mathbf{w})$ be the sample-wise estimate of the regularized risk. $\mathbf{w}$ is updated as follows:

$$\mathbf{w}^{(t)} = \mathbf{w}^{(t-1)} - \eta_t \nabla_{\mathbf{w}=\mathbf{w}^{(t-1)}} R(z_t; \mathbf{w}) \tag{5}$$

where η_t is the step size.

We provide in Table 1 the sampling and update procedures for the objective functions used in our evaluation, assuming no reweighting of the data. For R_{OVR}, R_{MUL} and R_{RNK}, these equations are straightforward and optimize exactly the regularized risk. For R_{WAR} it is only approximate as it does not compute exactly the value of $r_\Delta(\mathbf{x}_i, y_i)$, but estimates it from the number of samples k which were drawn before a violating sample $\bar{y}$ such that $L_{\text{tri}}(x_i, y_i, \bar{y}; \mathbf{w}) > 0$ was found. If k samples were drawn, then: $r_\Delta(\mathbf{x}_i, y_i) \approx \lfloor \frac{C-1}{k} \rfloor$. For a large number of classes, this approximate procedure is significantly faster than the exact one (see [38] for more details). Note also that we implement the regularization by penalizing the squared norm of $\mathbf{w}$, while [38] actually bounds the norm of $\mathbf{w}$. We tried both strategies and observed that they provide similar results in practice.

4.2. Implementation details

We use as basis for our code the SGD library for binary classification available on Bottou's website (version 1.3)[7]. This is already an optimized code which includes fast linear algebra and a number of optimizations such as the use of a scale variable to update $\mathbf{w}$ only when a loss is incurred. We now discuss a number of implementation details.

Bias. Until now, we have not considered the bias in our objective functions. This corresponds to an additional parameter per class. Following common practice, we add one constant feature to each observation $\mathbf{x}_i$. As is the case in Bottou's code, we do not regularize this additional dimension.

Stopping criterion. Since at each step in SGD we have a noisy estimate of the objective function, this value cannot be used for stopping. Therefore, in all our experiments, we use a validation set and stop iterating when the accuracy does not increase by more than a threshold θ.

Regularization. While a vast majority of the works on large-margin classification regularize explicitly by penalizing the squared norm of w (or by bounding it), regularizing implicitly by early stopping is another option, *i.e.* one sets $\lambda = 0$ and iterates SGD until the performance converges on the validation set (see *e.g.* [2]). In our experiments, applying this strategy yields competitive results.

Step size. To guarantee converge to the optimum, the sequence of step sizes should satisfy $\sum_{t=1}^{\infty} \eta_t = \infty$ and $\sum_{t=1}^{\infty} \eta_t^2 < \infty$. Assuming $\lambda > 0$, the usual choice is $\eta_t = 1/\lambda(t + t_0)$, where t_0 is a parameter to tune. Bottou provides in his code a heuristic to set t_0. We tried to cross-validate t_0 but never observed significant improvements. However, we also experimented with a fixed step size η as in [2, 38].

Reweighting. All sampling/update equations in Table 1 are based on non-reweighted objective functions (*c.f.* Section 3.2). To reweight the data, we can modify either the sampling or the update equations. We chose the first alternative since, in general, it led to faster convergence. If we take the example of L_{OVR} then the sampling is modified as follows: draw $y = +1$ with proba ρ, $y = -1$ with proba $1 - \rho$. Then draw x_i such that $y_i = y$.

4.3. Comparison with batch solvers

It is well known that SGD can perform as well as batch solvers for OVR SVMs at a fraction of the cost [7, 8]. However, to the best of our knowledge, public SGD solvers do not exist for other SVM formulations such as MUL SVM. As a sanity check, we compared our MUL SGD solver to two MUL batch solvers: LibLINEAR [13] and SVM-Light [17]. Because of the cost of running batch solvers, we ran experiments only on small subsets of ImageNet on small BOV vectors. Our experiments (not reported here) show that our SGD solver can perform on par with these solvers at a fraction of the cost.

	Sampling	Update
R_{OVR}	Draw $(\mathbf{x}_i, y_i)$ from S.	$\delta_i = 1$ if $L_{\text{OVR}}(\mathbf{x}_i, y_i; \mathbf{w}) > 0, 0$ otherwise. $\mathbf{w}^{(t)} = (1 - \eta_t \lambda)\mathbf{w}^{(t-1)} + \eta_t \delta_i \mathbf{x}_i y_i$
R_{MUL}	Draw $(\mathbf{x}_i, y_i)$ from S.	$\bar{y} = \arg\max_y \Delta(y_i, y) + \mathbf{w}_y' \mathbf{x}_i$ and $\delta_i = \begin{cases} 1 & \text{if } \bar{y} \neq y_i \\ 0 & \text{otherwise.} \end{cases}$ $\mathbf{w}_y^{(t)} = \begin{cases} \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) + \delta_i \eta_t \mathbf{x}_i & \text{if } y = y_i \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) - \delta_i \eta_t \mathbf{x}_i & \text{if } y = \bar{y} \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) & \text{otherwise.} \end{cases}$
R_{RNK}	Draw $(\mathbf{x}_i, y_i)$ from S. Draw $\bar{y} \neq y_i$ from $\mathcal{Y}$.	$\delta_i = 1$ if $L_{\text{tri}}(\mathbf{x}_i, y_i, \bar{y}; \mathbf{w}) > 0, 0$ otherwise. $\mathbf{w}_y^{(t)} = \begin{cases} \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) + \delta_i \eta_t \mathbf{x}_i & \text{if } y = y_i \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) - \delta_i \eta_t \mathbf{x}_i & \text{if } y = \bar{y} \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) & \text{otherwise.} \end{cases}$
R_{WAR}	Draw $(\mathbf{x}_i, y_i)$ from S. For $k = 1, 2, \ldots, C-1$, do: Draw $\bar{y} \neq y_i$ from $\mathcal{Y}$. If $L_{\text{tri}}(\mathbf{x}_i, y_i, \bar{y}; \mathbf{w}) > 0$, break.	$\delta_i = 1$ if $\bar{y}$ s.t. $L_{\text{tri}}(\mathbf{x}_i, y_i, \bar{y}; \mathbf{w}) > 0$ was sampled, 0 otherwise. $\mathbf{w}_y^{(t)} = \begin{cases} \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) + \delta_i \ell_{\lfloor \frac{C-1}{k} \rfloor} \eta_t \mathbf{x}_i & \text{if } y = y_i \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) - \delta_i \ell_{\lfloor \frac{C-1}{k} \rfloor} \eta_t \mathbf{x}_i & \text{if } y = \bar{y} \\ \mathbf{w}_y^{(t-1)}(1 - \eta_t\lambda) & \text{otherwise.} \end{cases}$

Table 1. Sampling and update equations for various objective functions.

5. Experiments

The experimental setup is described in Section 5.1. A detailed analysis of the different objective functions and parameters is given in Section 5.2 for the ILSVRC 2010 dataset. Section 5.3 presents results on the large-scale ImageNet10K dataset.

5.1. Experimental setup

Datasets. We use two subsets of ImageNet [12] for our evaluation. The ILSVRC2010 dataset [5] contains 1,000 classes and 1.4M images. This dataset was used in the ImageNet Large Scale Visual Recognition Challenge in 2010. We follow the standard training/validation/testing protocol (resp. 1.2M/50K/150K images). The ImageNet10K dataset contains 10,184 classes and approx. 9M images [11]. Following [29], we use half of the data for training, 50K images for validation and the remainder for testing. In all these experiments, we compute the top-1 (or top-5) accuracy per class and report the average [11, 29, 38].

Features. Images are resized to 100K pixels if larger. We extract approx. 10K SIFT descriptors [20] from 24x24 patches on a regular grid every 6 pixels at 5 scales. They are reduced from 128-dim to 64-dim using PCA. These descriptors are then aggregated into an image-level signature. We use the Fisher Vector (FV) representation which was shown state of the art in a recent evaluation [9]. By default we use $N = 256$ Gaussians and a spatial pyramid (SP) with $R = 4$ regions (the entire images and three horizontal stripes). The resulting descriptor is of approx. 130K dimensions. We also report results with the bag of visual words (BOV) given its popularity in large-scale classification [11, 38]. Our default BOV is 4K dimensional with $N = 1,024$ and $R = 4$. Given the large datasets we work with, image signatures need to be compressed. For FV we employ product quantization [15] and for BOV scalar quantization [11]. Signatures are decompressed on-the-fly by the SGD routines [29].

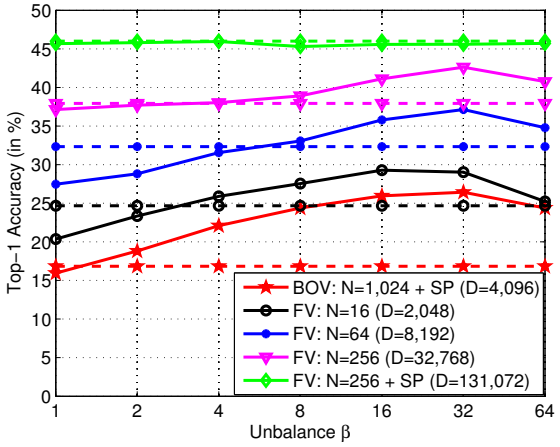


Figure 1. Impact of imbalance $\beta = (1 - \rho)/\rho$ on the accuracy on ILSVRC 2010. The plain lines correspond to w-OVR while the dashed lines correspond to u-OVR (which is independent of β). For each method, N is the number of Gaussians, SP indicates whether spatial pyramids were used and D is the dimensionality of the features. Similar curves were obtained for top-5 accuracy.

5.2. Detailed analysis on ILSVRC2010

Importance of reweighting in OVR. We first show the importance of reweighting the positive and negative samples in OVR. In what follows, u-OVR refers to the unweighted version (all samples have the same weight) while w-OVR refers to the reweighted version. Since we reweight by biasing the sampling (*c.f.* Section 3.2), we introduce the im-

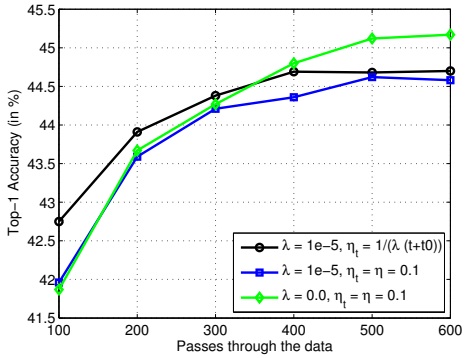


Figure 2. Impact of regularization on w-OVR (with $\beta = 1$). Results on ILSVR 2010 with 130K-dim FVs. One pass signifies seeing all positives for a given class + (on average) as many negatives. Hence, 500 passes is approximately as costly as seeing each sample of the dataset once (because there are 1,000 classes). Similar curves were obtained for top-5 accuracy.

balance parameter $\beta = (1-\rho)/\rho$ which is the (average) number of negatives sampled for each positive. $\beta = 1$ corresponds to the natural rebalancing of the data (giving the same weight to positives and negatives). We experimented with different image signatures and especially with different dimensions for FV by varying the number of Gaussians N and by using a SP or not. Results are reported in Figure 1. We can see that properly tuning β can have a significant impact on accuracy. Especially for smaller dimensional representations, w-OVR significantly outperforms u-OVR and the natural rebalancing (*i.e.* $\beta = 1$) is insufficient to get good results. For the largest FV, cross-validating β has little effect. While it had been shown that tuning this parameter is important to plot ROC curves in two-class classification [1], we believe we are the first to show that this parameter has a crucial impact for multiclass problems.

Note that we also reweighted the data for MUL, RNK and WAR but this had virtually no impact on accuracy. One of the reasons is that, although we work on the same data, it is much less unbalanced for MUL, RNK and WAR than for OVR. Indeed, the ratio between the number of samples in the most and least populated classes on ILSVRC2010 is approx. 4. In the OVR case however the imbalance for a given class is the ratio between the number of positive and negative samples and is therefore on the order of 999.

Implicit regularization works. The following experiments are reported for w-OVR with natural reweighting ($\beta = 1$), but similar results were obtained with different β parameters or with MUL, RNK and WAR. We ran experiments on the largest features (the 130K-dim FV), *i.e.* when regularization is supposed to have the largest impact. We compare three regularization strategies: (i) using explicit regularization ($\lambda > 0$) and a decreasing step size ($\eta_t = 1/(\lambda(t+t0))$, (ii) using explicit regularization and a fixed step size ($\eta_t = \eta$) and (iii) using no regularization ($\lambda = 0$) and a fixed step size, *i.e.* implicitly regularizing with the number of iterations. The test results are shown in Figure 2 for the best cross-validated parameters of the three methods. Strategy (i), which is the standard approach [7], is slightly faster to converge in the first iterations thanks to initial larger step sizes. However, at convergence all three methods yield similar results. A major advantage of strategy (iii) with respect to strategies (i) and (ii) is that there is a single parameter to tune (the number of iterations $niter$) instead of two ($niter$ and λ). Given the training times in large-scale (*c.f.* Section 5.3), we choose this approach in the following experiments. Note that we also experimented with the implicit regularization with early stopping on the small-scale PASCAL VOC 2007 dataset and observed a small drop of performance compared to explicit regularization (from 62.1% to 60.2%) which seems to indicate that this strategy is better suited to large-scale datasets.

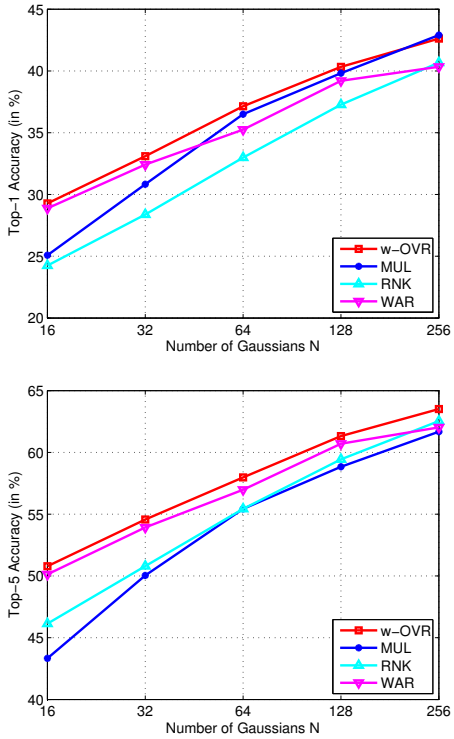


Figure 3. Impact of the number of Gaussians N (and therefore of the feature dimensionality) on the classification accuracy. Top: top-1. Bottom: top-5. Because of the cost of running these experiments, we did not use SPs.

One-vs-rest works. We compare OVR, MUL, RNK and WAR. For OVR, we report results for the re-weighted version, *i.e.* w-OVR. For MUL, RNK and WAR we report the results for the unweighted versions only (again the difference with the weighted versions was typically on the order

		w-OVR	MUL	RNK	WAR
Top-1	BOV	26.4	22.7	20.8	24.1
	FV	45.7	46.2	46.1	46.1
Top-5	BOV	46.4	38.4	41.2	44.2
	FV	65.9	64.8	65.8	66.5

Table 2. Accuracy (in %) on ILSVRC 2010.

of 0.1-0.5%). We report results for 130K-dim FVs and 4K-dim BOV vectors in Table 2. We can draw the following conclusions.

First, for high-dimensional FV features all methods perform pretty much the same: the difference between the best and worst performing methods is 0.5% at top-1 and 1.7% at top-5. We performed an additional set of experiments on FVs to study the influence of the signature dimensionality. The results are reported in Figure 3. We can indeed observe that, as the number of Gaussians N increases, *i.e.* as the features grow larger, the difference of accuracy between different methods becomes smaller. Hence, for high-dimensional features, the learning objective function seems to have little impact. Put into the perspective of the statistical learning theory described in [3], this implies that the $\psi(\cdot)$ function associated to each surrogate loss – which defines how the performance in the surrogate loss transfers to the target loss – does not play a major role here (see Theorem 10 in [3]). Indeed, in our experiments the different learning objectives corresponding to different surrogate losses lead to similar, yet, good results. Therefore, in these experiments, the capacity of the classifiers at hand is "large" enough, relative to the difficulty of the dataset, and this corresponds to the so-called "low-noise" condition in statistical learning under which ones gets "fast learning rates". In this case, the differences between the surrogate losses almost disappear in the fast rate regime.

Second, w-OVR seems to always perform best closely followed by WAR. As expected, MUL which focuses on the first rank seems to perform better at top-1 than at top-5 while RNK, which optimizes the rank of the correct labels, seems to be more competitive for top-5. An important conclusion is that, despite its simplicity and its supposed suboptimality, OVR is a competitive alternative to more complex objective functions on ILSVRC 2010.

Comparison with the state of the art. We note that the two winning teams at ILSVRC 2010 reported better results than our 66.5% at top-5. [29] reports 74.3% by combining SIFT and color descriptors and by using 520K-dim FVs while [19] reports 71.8% by combining SIFT and LBP descriptors as well as multiple encoding techniques and spatial pyramids. While better features can indeed increase accuracy, this is out of scope here, as we focus on learning the classifier.

	u-OVR	w-OVR	MUL	RNK	WAR
BOV 4K-dim	3.8	7.5	6.0	4.4	7.0
FV 130K-dim	-	19.1	-	-	17.9

Table 3. Top-1 accuracy (in %) on ImageNet10K

5.3. Large-scale experiments on ImageNet10K

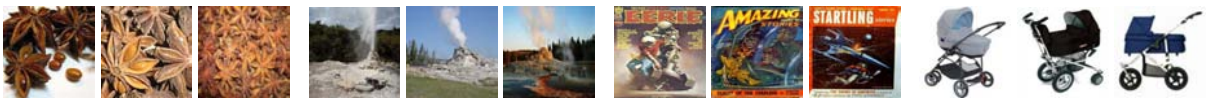
We now summarize our experimental results on the large ImageNet10K dataset. In Table 3, we report results for the 4K-dim BOV and the 130K-dim FV using top-1 accuracy as in [11, 29]. Given the cost of running the experiments on the high-dimensional FVs, we carried-out experiments only with the two objective functions which performed best on ILSVRC 2010: w-OVR and WAR. Again, our conclusion is that w-OVR performs better than more complex objective functions (at least on this dataset with those features). See Figure 4 for sample results.

Comparison with the state of the art. We now compare our results with publications which report results at the scale of $O(10^4)$ categories [11, 29, 38]. Compared to [11], our BOV results are on par (even slightly better since we report 7.5% while they report 6.4%) and our FV results are significantly (almost 3 times) better due to the use of higher-dimensional features.

Weston *et al.*, who used a different ImageNet subset in their experiments, show that WAR outperforms OVR [38] on BOV descriptors, where their OVR baseline did not reweight the positives/negatives, *i.e.* it is similar to our u-OVR. We also observed that WAR significantly outperforms u-OVR. However, we showed experimentally that w-OVR performs significantly better than u-OVR and slightly better than WAR.

As for [29], Sánchez and Perronnin use w-OVR with natural rebalancing ($\beta = 1$). We show that properly tuning β can have a significant impact on accuracy: using the same features, we improve their baseline by an absolute 2.4%, from 16.7% to 19.1%. It is interesting to note that while rebalancing the data had little impact on the 130K-dim FV on ILSVRC 2010, it has a significant impact on ImageNet10K. This does not contradict our previous statement that different objective functions perform similarly on high-dimensional features. Features are only high-dimensional with respect to the complexity of the problem and especially the number of classes. While 130K-dim is high-dimensional with respect to the 1K categories of ILSVRC 2010, it is not high-dimensional anymore with respect to the 10K categories of ImageNet10K.

Timings for ImageNet10K and 130K-dim FVs. For the computation we used a small cluster of machines with 16 CPUs and 32GB of RAM. The feature extraction step (including SIFT description and FV computation) took approx. 250 CPU days, the learning of the w-OVR SVM approx.



(a) Star Anise (92.45%) (b) Geyser (85.45%) (c) Pulp Magazine (83.01%) (d) Carrycot (81.48%)

(e) European gallinule (15.00%) (f) Sea Snake (10.00 %) (g) Paintbrush (4.68 %) (h) Mountain Tent (0.00%)

Figure 4. ImageNet10K results (top-1 accuracy in %) obtained with w-OVR and 130K-dim Fisher vectors. (a-d) Sample classes among the best performing ones. (e-h) Sample classes among the worst performing ones.

400 CPU days and the learning of the WAR SVM approx. 500 CPU days. Note that w-OVR performs slightly better than WAR and is much easier to parallelize since the classifiers can be learned independently.

References

[1] F. Bach, D. Heckerman, and E. Horvitz. Considering cost asymmetry in learning classifiers. *JMLR*, 2006. 6

[2] B. Bai, J. Weston, D. Grangier, R. Collobert, O. Chapelle, and K. Weinberger. Supervised semantic indexing. In *CIKM*, 2009. 4

[3] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. In *NIPS*, 2003. 7

[4] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *NIPS*, 2010. 2

[5] A. Berg, J. Deng, and L. Fei-Fei. ILSVRC 2010. http://www.image-net.org/challenges/LSVRC/2010/index. 1, 5

[6] A. Bordes, L. Bottou, P. Gallinari, and J. Weston. Solving multiclass support vector machines with LaRank. In *ICML*, 2007. 1

[7] L. Bottou. SGD. http://leon.bottou.org/projects/sgd. 4, 6

[8] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *NIPS*, 2007. 1, 4

[9] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, 2011. 1, 5

[10] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *JMLR*, 2001. 1, 3

[11] J. Deng, A. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? In *ECCV*, 2010. 1, 2, 5, 7

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 2, 5

[13] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 2008. 4

[14] T. Gao and D. Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *ICCV*, 2011. 2

[15] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE TPAMI*, 2011. 5

[16] T. Joachims. Optimizing search engines using clickthrough data. In *SIGKDD*, 2002. 1, 3

[17] T. Joachims. Training linear SVMs in linear time. KDD, 2006. 4

[18] Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. In G. Orr and M. K., editors, *Neural Networks: Tricks of the trade*. Springer, 1998. 1

[19] Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, and T. Huang. Large-scale image classification: Fast feature extraction and SVM training. In *CVPR*, 2011. 1, 2, 4, 7

[20] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2), 2004. 5

[21] S. Maji and A. Berg. Max-margin additive classifiers for detection. In *ICCV*, 2009. 2

[22] M. Marszalek and C. Schmid. Constructing category hierarchies for visual recognition. In *ECCV*, 2008. 2

[23] S. Nowozin and C. Lampert. Structured learning and prediction in computer vision. *Foundations and Trends in Computer Graphics and Vision*, 2011. 1

[24] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007. 1

[25] F. Perronnin, J. Sánchez, and Y. Liu. Large-scale image categorization with explicit data embedding. In *CVPR*, 2010. 2, 4

[26] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *ECCV*, 2010. 1, 4

[27] R. Rifkin and A. Klautau. In defense of one-vs-all classification. *JMLR*, 2004. 2

[28] M. Rohrbach, M. Stark, and B.Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, 2011. 1, 2, 4

[29] J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *CVPR*, 2011. 1, 2, 4, 5, 7

[30] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimate sub-gradient solver for SVM. In *ICML*, 2007. 1

[31] A. Tewari and P. L. Bartlett. On the consistency of multiclass classification methods. *JMLR*, pages 1007–1025, 2007. 3

[32] A. Torralba, R. Fergus, and W. Freeman. 80 million tiny images: a large dataset for non-parametric object and scene recognition. *IEEE PAMI*, 2008. 2

[33] L. Torresani, M. Szummer, and A. Fitzgibbon. Efficient object category recognition using classemes. In *ECCV*, 2010. 2

[34] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 2005. 3

[35] N. Usunier, D. Buffoni, and P. Gallinari. Ranking with ordered weighted pairwise classification. In *ICML*, 2009. 1, 2, 3

[36] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. In *CVPR*, 2010. 2

[37] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, 2010. 1

[38] J. Weston, S. Bengio, and N. Usunier. Large scale image annotation: Learning to rank with joint word-image embeddings. *ECML*, 2010. 1, 2, 3, 4, 5, 7

[39] J. Weston and C. Watkins. Multi-class support vector machines. Technical report, Department of Computer Science, Royal Holloway, University of London, 1998. 3

[40] J. Xu, T. Liu, M. Lu, H. Li, and W. Ma. Directly optimizing evaluation measures in learning to rank. In *SIGIR*, 2008. 1, 3

[41] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *SIGIR*, 2007. 1, 3

[42] Z. Zhou, K. Yu, T. Zhang, and T. Huang. Image classification using super-vector coding of local image descriptors. In *ECCV*, 2010. 1