JMLR: Workshop and Conference Proceedings 37:1–16, 2014 ACML 2014

Sample Distillation for Object Detection and Image Classification

Olivier Canévet OLIVIER.CANEVET@IDIAP.CH
Leonidas Lefakis LEONIDAS.LEFAKIS@IDIAP.CH
François Fleuret FRANCOIS.FLEURET@IDIAP.CH
Computer Vision and Learning group, Idiap Research Institute, Martigny, Switzerland
École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

Editor: Dinh Phung and Hang Li

Abstract

We propose a novel approach to efficiently select informative samples for large-scale learning. Instead of directly feeding a learning algorithm with a very large amount of samples, as it is usually done to reach state-of-the-art performance, we have developed a "distillation" procedure to recursively reduce the size of an initial training set using a criterion that ensures the maximization of the information content of the selected sub-set.

We demonstrate the performance of this procedure for two different computer vision problems. First, we show that distillation can be used to improve the traditional bootstrapping approach to object detection. Second, we apply distillation to a classification problem with artificial distortions. We show that in both cases, using the result of a distillation process instead of a random sub-set taken uniformly in the original sample set improves performance significantly.

Keywords: Greedy Edge Expectation Maximization (GEEM), Boosting, Object detection, Image classification

1. Introduction

Image classification and object detection have reached an acceptable level of performance thanks to the use of machine learning techniques with large-scale training sets. In both cases, large amounts of data points can be produced, either through artificial distortions of positive samples, or through the bootstrapping of negative samples, the objective being to enrich the available population, either in the neighborhood of positive examples, or at the interface between the positive and the negative population, as characterized by a predictor trained on a limited data-set.

In such cases, the crux of the problem is the training itself, based on these large amounts of data. Most of the learning algorithms have a computation cost at least linear with the number of samples, and are difficult to parallelize.

Interestingly, while most training sets are redundant, very few methods explicitly try to leverage that redundancy to reduce the computational cost. Some well known techniques such as the stochastic gradient descent are explicitly justified through the redundancy of samples, but no method exists that processes the data in the same spirit as a feature-

© 2014 O. Canévet, L. Lefakis & F. Fleuret.

selection procedure does with a feature space: By explicitly reducing the cardinality of the space, while keeping the informative content as high as possible.

We propose in this article to exploit a recent machine-learning technique called "reservoir learning" by Lefakis and Fleuret (2013) which has been developed precisely to select jointly informative sub-sets of samples. We adapt it to a large-scale context by applying it in a recursive manner, allowing to reduce the overall computation cost by several orders of magnitude, to control its memory footprint, and to parallelize it on a multi-core architecture.

We apply this "distillation" procedure to Boosting in two different contexts: Pedestrian detection in natural images, where it allows to improve the set of bootstrapped negative samples, and character recognition, where distillation is used to get a better set of synthetic distortions.

2. Related works

Though bootstrapping is widely used in computer vision (Shotton et al., 2005), there has been very little prior work on how to efficiently mine hard negative examples. Typically once a classifier has been built, it is used to process the set of negative samples and a margin based approach is used to select which of these samples are to be added to the training set (Felzenszwalb et al., 2010; Li et al., 2013). Other approaches are possible, as for example using non-maxima suppression (Neubeck and Van Gool, 2006; Comaniciu et al., 2002); non-maxima suppression is also widely used to handle overlapping detections but despite its prevalence there seems to be no prior experimentation with this method for hard negative sample mining. We present such results in our experiments as a baseline for our proposed method.

A different approach to representing the richness of the background class is to attempt to model it using a (Normal) distribution (Osadchy et al., 2012; Hariharan et al., 2012). Negative samples can then be generated using a fitted model. Such approaches however tend to be quite slow and ignore the fact that natural images tend to have long-tailed distributions as shown by Ruderman and Bialek (1993).

A notable exception to this scarcity is the recent work by Henriques et al. (2013), that specifically addresses hard negative mining, or to be exact how to avoid it. The authors present a formulation for a specific family of classifiers that allows them to address the translations of samples in the negative sample space by modeling these translations as circular shifts. Though attaining very good results, their method is specific to translations while the method presented here handles the entire negative sample space. Furthermore, their approach cannot be used in connection with the boosting family of predictors.

The other method commonly employed in vision applications, is that of augmenting the positive sample space by random distortions of its population, sometimes referred to as jittering. This technique is used to artificially augment the number of positive samples which is oftentimes limited due to the cost of labeling. Distortions techniques used can be mirror versions of the originals, affine transformations (translations, scaling) like in the work of LeCun et al. (1998) and even elastic distortions in the work of Ciresan et al. (2012). Here there has been even less prior work. Distortions are usually generated at random and no care is taken to choose informative ones.

In general the problem of sub-sampling data has received little attention from the vision community despite its applicability. Given the increase of the data availability, we expect this problem to become more and more central for computer vision as it has become for machine learning.

In fact practically all prior work on the problem of data subsampling comes from the machine learning community, having received little attention in computer vision, Srinivasan and Duraiswami (2009) being a notable exception. In the context of boosting (Freund and Schapire, 1997) however there have been many approaches proposed for dataset subsampling (Bradley and Schapire, 2007; Domingo and Watanabe, 2000). Recently Lefakis and Fleuret (2013) have proposed a novel algorithm for addressing this issue, which aims to select the most informative samples, in a joint manner, amongst a large number of samples. Their work focuses on trade-offs between online and offline learning, here however we aim to extend this work so as to tackle the, computer vision focused, sample set selection problems mentioned above.

3. Proposed Method

We first present a brief overview of boosting and the Greedy Edge Expectation Maximization (GEEM) algorithm proposed by Lefakis and Fleuret (2013) and subsequently present and analyze the proposed distillation method. We leave the specifics of its application to the experimental part of the paper.

3.1. Boosting and GEEM

Boosting algorithms greedily build an ensemble classifier H

$$H(x) = \sum_{t=1}^{T} a_t h_t(x) \tag{1}$$

by adding what is referred to as a weak learner $h_t(x) \in \{-1, 1\}$ at each iteration t among a pool of weak learners $\mathscr{H}$ so as to minimize an empirical loss

$$h_t = \arg\min_{h \in \mathscr{H}} L_{\mathscr{T}}(H_{t-1} + h_t) \tag{2}$$

with $\mathscr{T}$ being the training data.

As mentioned, a problem that often arises when handling large datasets is that the size of $\mathscr{T}$ is such that it does not all fill in memory making the optimization problem 2 unwieldy or even intractable.

Typically in such cases the training set $\mathscr{T}$ is subsampled at each iteration t to yield a subset $\mathscr{T}^t \subset \mathscr{T}$ which is then used to acquire an estimate of the empirical loss.

The GEEM algorithm of Lefakis and Fleuret (2013) addresses the issue of optimally sampling $\mathscr{T}^t$ under certain assumptions. Specifically the authors assume that the weak learner responses on the samples follow a multivariate Gaussian distribution, they then use a greedy backward selection procedure to decide which samples to keep in $\mathscr{T}^t$, so that the expected empirical loss on $\mathscr{T}$, when using $\mathscr{T}^t$ to solve (2), is minimal.

In other words, given a signed (boosting) weight for each sample, the GEEM procedure selects a subset of samples such that, if a predictor has a large weighted response over the selected samples, it has a large expected weighted response over the selected samples *and* the discarded ones. The GEEM selection process accounts for correlation between predictor responses, hence allowing to discard samples whose response vectors are redundant.

3.2. Distillation

While GEEM has many desirable properties, and despite the efficient implementation proposed in Lefakis and Fleuret (2013), it remains extremely costly, with the cost being prohibitive for large amounts of data, exactly in the regime that is of interest to modern applications, due to its cubic dependency on the number of samples N. We propose to drastically reduce this cost, while maintaining the statistical accuracy, by processing the data recursively using a tree structure.

Given the full set of N samples $\mathscr{T}^*$, we first partition it into K disjoint subsets[1]

$$T_1^0, \ldots, T_K^0, \ \forall k \, |T_k| = \frac{N}{K},$$

We then apply the GEEM procedure recursively, each time merging two subsets T_{2r-1}^d and T_{2r}^d and extracting one T_r^{d+1} of half the cardinality:

$$T_r^{d+1} \subset T_{2r-1}^d \cup T_{2r}^d, \text{ s.t. } |T_r^{d+1}| = \frac{N}{K},$$

until only one subset of samples T_1^D remains.

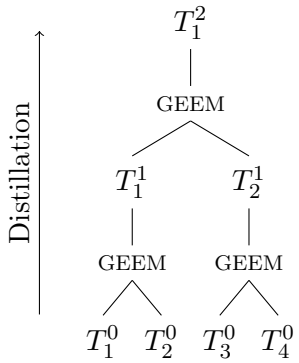


Figure 1: The distillation process consists in starting from a partition $T_1^0, \ldots, T_K^0$ (with $K = 4$ here for simplicity) of the full set of samples $\mathscr{T}$, then recursively concatenating sub-sets by pairs, and extracting one subset of fixed size $\frac{N}{K}$ from each resulting set using the GEEM procedure.

1. Here the superscript denotes the tree level of a node in the tree (and by extension the subset it contains) and the subscript denotes the numbering of the node at its level.

3.3. Complexity and trade-offs

Lefakis and Fleuret (2013) analyze the complexity of GEEM and point out the limitations on the applicability of the algorithm due to its cubic complexity. By using the tree structure presented in the previous subsection, this cubic complexity becomes less of a restraining factor.

Whereas in reservoir Boosting the method's complexity is $O(N^3)$, N being the number of samples, in distillation the cost is $O(N^3/K^2)$ where K is the number of tree leaves. For a balanced binary tree with K leaves there are $O(K)$ nodes in the tree, each containing $O(N/K)$ samples. Thus sampling between two nodes incurs a cost of $O(N^3/K^3)$ and this process must be repeated $O(K)$ times, leading to the $O(N^3/K^2)$ complexity which translates to a K^2 complexity improvement.

Moreover the tree structure allows a significant degree of parallelization. The sampling in a specific node of the tree can obviously be performed independently with the remaining nodes at the same level of the tree. Thus, depending on the number of CPU cores available, one can choose a trade-off between computation time and memory load. In contrast, in the case of reservoir Boosting, the GEEM process cannot be similarly parallelized.

Figure 2 shows the two extreme cases of computation-memory trade-off in the distillation process. In the first case, there is only one CPU core available, and the main concern is the memory load. A naive implementation would first load all samples in the leaves and proceed up the tree level by level. Depending on the size of N this procedure might be prohibitive from a memory perspective.

The tree however can be traversed in a much more memory-efficient manner, as depicted in Figure 2(a) by always fully processing any sub-tree before starting to process a leaf belonging to another one.

Let $M(K)$ be the memory footprint of the distillation of K leaves. Obviously $M(1) = 1$, since there is nothing to do and we must have the samples of that leaf in memory. If we have to distill $2L$ leaves, we can first distill the first L, with a memory usage of at most $M(L)$, store the result for a memory usage of 1, and process the second half, for a memory usage of $M(L)$. Hence $M(2L) = 1 + M(L)$, which leads to $M(K) \sim O(\log_2 K)$

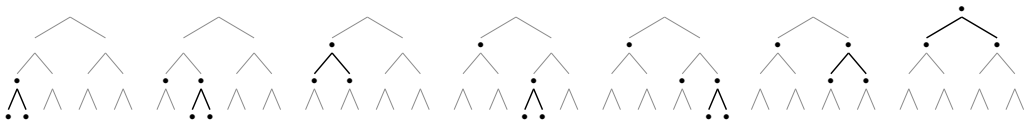
Computation-wise, since we have to run sequentially through all the nodes, the computation time is $O(K)$

In the case of K cores, shown in Figure 2(b), we can achieve a computation of $O(\log_2 K)$ by assigning each node in a level to a different core and proceeding upward in a breadth-first manner, i.e. first we process all nodes at one level before moving up to the next. Given that a balanced binary tree with K leaves has a height of $\log_d K$, the algorithm will terminate in time $O(\log_2 K)$. As pointed out above, loading the entire data at the leaves of the tree has a $O(K)$ memory footprint.

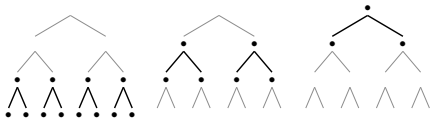
Thus according to the resources at hand, the distillation's tree structure allows us to choose the optimal trade-off between the two resource types.

4. Pedestrian detection

We first apply the distillation procedure to bootstrapping negative samples for object detection.



(a) Distillation on a single computation core



(b) Distillation on multiple computation cores

Figure 2: Comparison of the computation cost and memory footprint of the single-core distillation (top) and the multi-core one (bottom). Black dots picture batches of samples actually stored in memory, and bold edges the individual GEEM operations. If K is the total number of sample batches we have initially–that is the number of leaves in the distillation tree–and we have a single computation core, the memory footprint is at most $O(\log_2 K)$ (see §3.3) and the computation cost $O(K)$. If we have K computation cores, the memory footprint is at most $O(K)$ and the computation $O(\log_2 K)$.

4.1. Data-set

We use the INRIA person data-set (Dalal and Triggs, 2005) which is commonly used to train and evaluate pedestrian detectors. For training, the data consists of 2,416 64×128 cropped color images of pedestrians and a set of 1,218 pedestrian-free scenes. To evaluate the performances, the test set consists of 1,132 pedestrians and 453 scenes. The scenes are used to extract the negative and the false positive samples in the first training and in the bootstrapping phase. We use histograms of oriented gradients (HOG, Dalal and Triggs 2005; Felzenszwalb et al. 2010) as a descriptor for the images with the parameter values specified in Dalal and Triggs (2005).

4.2. Bootstrapping

Let $\mathscr{T}_+$ be the set of all available positive samples in $\mathscr{T}$, and $\mathscr{T}_-^*$ the set of all available negative samples. In the case of detection, there are usually a few thousands samples in the former, and millions or more in the latter.

In bootstrapping, we first train an initial two-class predictor f^0 with $\mathscr{T}_+$ and a subset $\mathscr{T}_-^0 \subset \mathscr{T}_-^*$, obtained by sampling uniformly in $\mathscr{T}_-^*$ without replacement.

The objective of both our distillation method and all the baselines is to use f^0 to select a subset $\mathscr{T}_-^1 \subset \mathscr{T}_-^*$ such that we can build a second predictor f^1 with $\mathscr{T}_+$ and $\mathscr{T}_-^1$, more accurate than f^0.

In all the techniques we consider, the number of selected negative samples N_- is a constant fraction $\zeta \in [0, 1]$ of the total number of false positives:

$$N_- = |\mathscr{T}_-^1| = \left\lceil \zeta \left| \left\{ x \in \mathscr{T}_-^*, \text{ s.t. } f^0(x) \geq 0 \right\} \right| \right\rceil$$

where $\lceil \, \rceil$ is the ceiling function (i.e. smallest integer greater than).

4.3. Performance evaluation

We compare our approach of selecting negative samples among the false positives with the following methods. The algorithms described below are applied per-scene.

- **Uniform sampling** consists in sampling N_- samples uniformly without replacement in the subset of false positives, that is, according to the distribution

  $$\forall x \in \mathscr{T}_-^*, \; p_x = \frac{1}{Z} \mathbf{1}_{\{f^0(x) \geq 0\}},$$

  where Z is a normalization constant.

- **Max-sampling** consists in selecting in $\mathscr{T}_-^*$ the N_- samples with highest f^0 responses, that is the ones which are the most "incorrectly classified":

  $$\forall x \in \mathscr{T}_-^1, \, x' \in \mathscr{T}_-^* \setminus \mathscr{T}_-^1, \; f(x) \geq f(x').$$

- **Weighted sampling** consists of sampling N_- samples in $\mathscr{T}_-^*$, without replacement, according to the exponential loss, that is with the distribution

  $$\forall x \in \mathscr{T}_-^*, \; p_x = \frac{1}{Z} \mathbf{1}_{\{f^0(x) \geq 0\}} e^{f^0(x)},$$

  where Z is a normalization constant. This method is somewhat similar to the max-sampling but also allows to pick samples of small individual weights which have a large cumulated weight together.

- **Non-maxima suppression** consists in selecting the samples with the highest f^0 responses and then discarding an area around the selected one to prevent from sampling in its neighborhood again. This method is used in object detection to remove multiple detections and in this case will tend to pick samples equally spaced in the image.

Figure 3(*a*) shows the receiver operating characteristic (ROC) curves of the trained predictors.

The initial predictor is trained with $P = 2,416$ positive images and $N = 10,000$ negative ones. We then retain 3% of the false positive per scene (or at least 10). which lead on average to $H \simeq 18,000$ hard negative samples. Both predictors are trained using AdaBoost for 1,000 rounds on the HOG features. The curves are averaged over five runs.

The predictor trained using **max-sampling** performs worse than the initial one. The **weighted sampling** method also performs worse than the initial predictor for the same reason but does better than max-sampling because it is still able to sample among false positive with smaller weights. The **uniform sampling** and the **non-maxima suppression** perform comparatively better than the initial predictor. These methods are able to populate the training set with difficult samples so that the Boosting procedure focuses on the interface between positive and negative classes. Finally, the **distillation method** does even better than the uniform sampling. Distillation is able to discard the false positive samples that do not bring anymore information to the predictor given the ones it has already picked. We can thus say that it has built a richer training set.

Figure 3(*b*) depicts the influence of K in the distillation process. As all the scenes do not contain the same number of false positives, fixing K accross all scenes would produce subsets of different sizes. Instead, we fix N/K (size of the subset) to be constant. As $N/K \to N$, the distillation gets closer to the true GEEM. The results show here that surprisingly, the size does not really matter. However, we have notice that to enforce good diversity in the final distilled set, scenes should be processed separately.

5. Character recognition

We now apply the GEEM procedure to enrich a training set with informative distortions for image classification.

5.1. Data-Set

We use the MNIST data-set (LeCun et al., 1998). The train set consists of 60,000 28×28 gray scale images of digits from 0 to 9 and the test set contains 10,000 images. We use Haar like wavelets (Viola and Jones, 2004) as features without pre-processing the images, as in the work of Kégl and Busa-Fekete (2009) where AdaBoostMH achieves 1.02% test error.

5.2. Sub-sampling distortions

In the case of jittering we seek to augment a relatively small set of positive samples $\mathscr{T}_+$ by randomly distorting an image and adding the resulting image to the pool of positive samples.

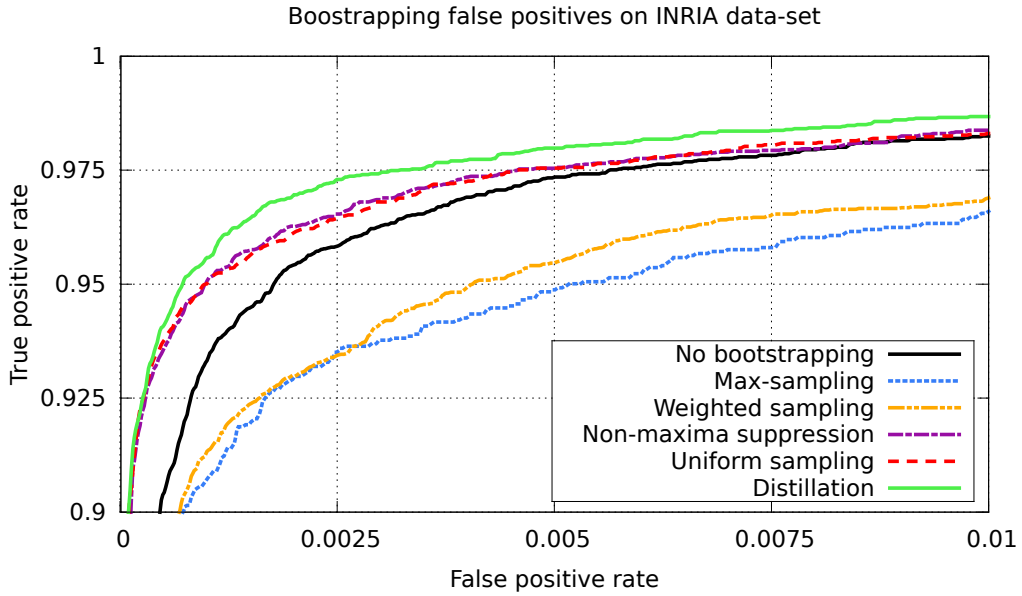
Suppose we create such a set of artificial positive samples $\mathscr{T}_+^{'}$, we can either create a set of small cardinality $|\mathscr{T}_+^{'}| = N_+$ or we can create multiple distortions resulting in a set of large cardinality and then downsample to N_+, using GEEM.

Again we first train an initial two-class predictor f^0 with $\mathscr{T}_+$ and $\mathscr{T}_-$ and then create a set $\mathscr{T}_+^{'}$ of cardinality $N^{'} \gg N_+$. We then use f_0 in connection with the GEEM process to subsample $\mathscr{T}_+^{'}$ down to $\mathscr{T}_+^1$.

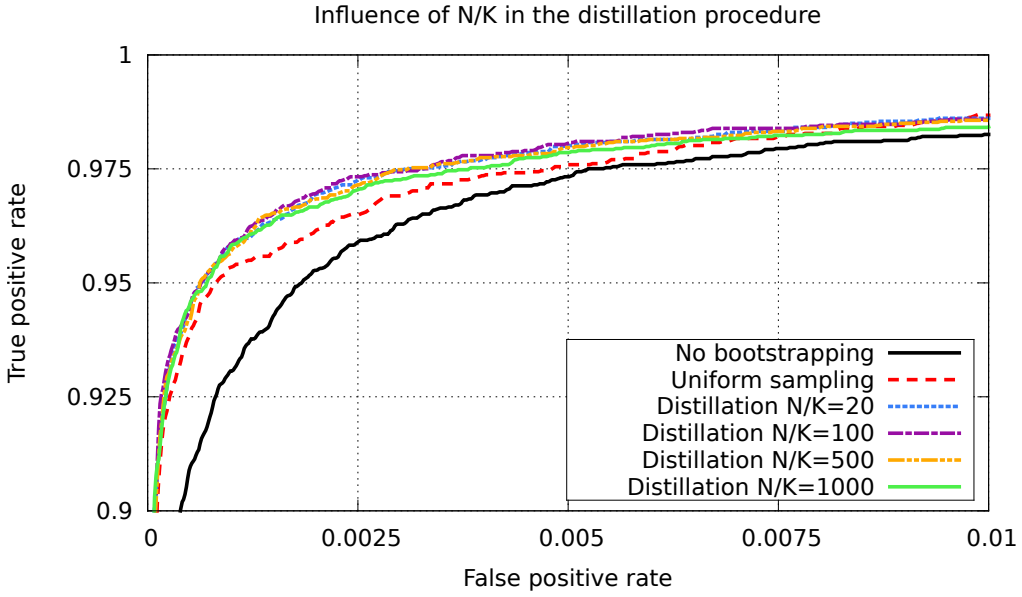
We then use $\mathscr{T}_+^1$ together with $\mathscr{T}_+$ and $\mathscr{T}_-$ to build the predictor f^1.

5.3. Performance evaluation

The images are distorted using the combination of translations of 1 pixel and rotation of angle $\alpha \in [-10, 10]$. Using higher shifts or angles produces images which are too far from the true distribution of images. Test images are not distorted at test time.



(a) Performance of the classifier



(b) Influence of N/K

Figure 3: Bootstrapping images to enrich the training set with difficult examples – On figure 3(a), the max-sampling and weighted sampling perform worse than no bootstrapping because they tends to put emphasis on redundant samples. The uniform sampling and the non-maxima suppression perform the same and our distillation process outperforms all other sampling methods. Figure 3(b) shows the influence of the size of the subsets in the distillation. The greater the size, the closer to the true GEEM the process is.

Table 1: Classification error on the full MNIST data-set – The GEEM procedure is once again able to choose better sample in comparison to random. We can make the same observation as for the reduced sets: adding too much GEEM samples does not do better than random (here more than 3)

Method	Test error (%)
No distortions	1.2* (+0.0361)
+1 random distortion per image	0.962 (± 0.0616)
+1 distilled distortion per image	0.9075 (+0.0585)
Explanation of AdaboostMH	

*To compute the error rate, Kégl and Busa-Fekete (2009) used AdaBoostMH whereas we use 1 vs. all AdaBoost. Also, they compute their error rate by averaging over the last 50,000 rounds of Boosting out of 100,000, while we use only 10,000 rounds. This explains the difference with our result.

As GEEM requires a initial classifier we train one on the non-distorted set. In our experiments the training set is augmented by n distortions per digit, that is for a set of S images and n distortions per image, we train the second classifier with $(n+1)S$ images.

We first analyze the performances of the GEEM procedure on a reduced training set of 1,000 and 5,000 images. Results for full scale experiments are shown in table 1.

Figure 4 shows that the GEEM procedure is able to build a more informative training set than the random. The test accuracy is higher.

The reason lies in how the Boosting loss is reduced with this GEEM. Figure 5 show the loss of validation samples in the early iterations of the Boosting algorithm when training with the new augmented training set. The validation samples were chosen among the discarded distortions for the GEEM.

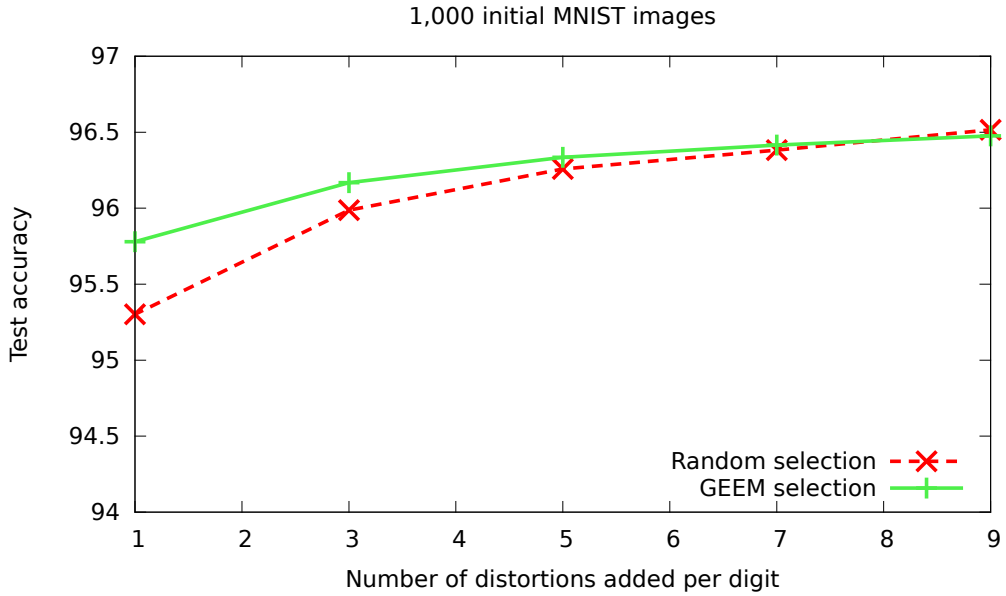
5.4. Analysis of the GEEM behavior

To understand a bit further why GEEM is able to build a more informative training set, we analyze its behavior in specific settings.

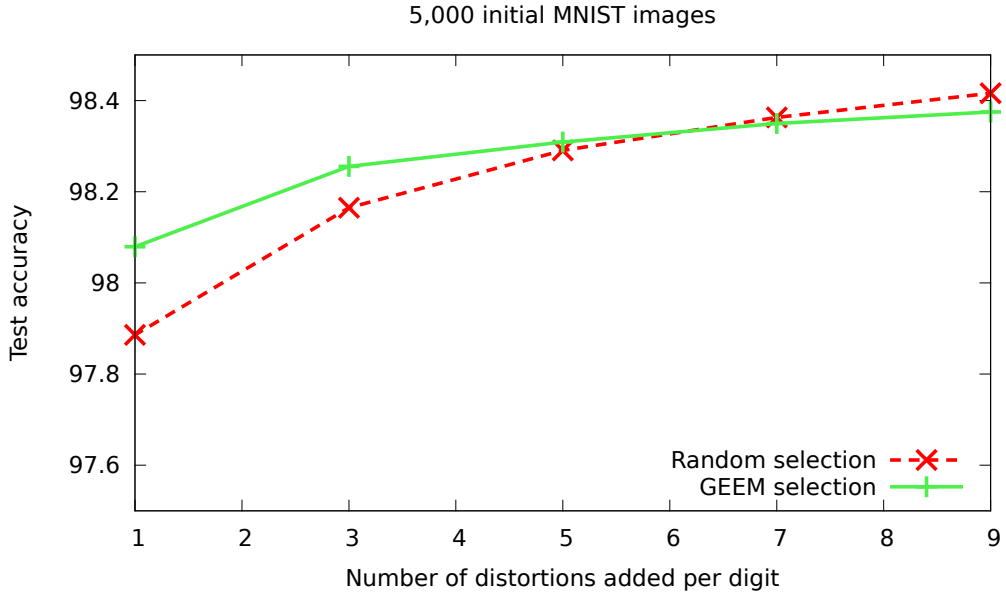
5.4.1. How does GEEM handle similar images?

To analyze how GEEM behave with similar images, we select 5 images from the same class and duplicate the first one 124 times (in this set, images from index 1 to 124 are the same, and images 125 to 128 are all different). We run the GEEM procedure to discard images down to 6 and note which indexes are kept. We run this experiment 1,000 times, each time selecting a new class and 5 images of the class and count how many times each index was kept by the process.

The histogram in Figure 6 shows how many times the images were selected on average over the 1,000 runs. We see that the last 4 indexes (i.e. the images which were unique in the set) tend to be selected more often than any of the duplicated one. Since we select 6 images out of 5 unique ones, the duplicated one is at least selected twice at each round. The GEEM procedure is therefore able to build a diverse set of samples that can be used for training another classifier.

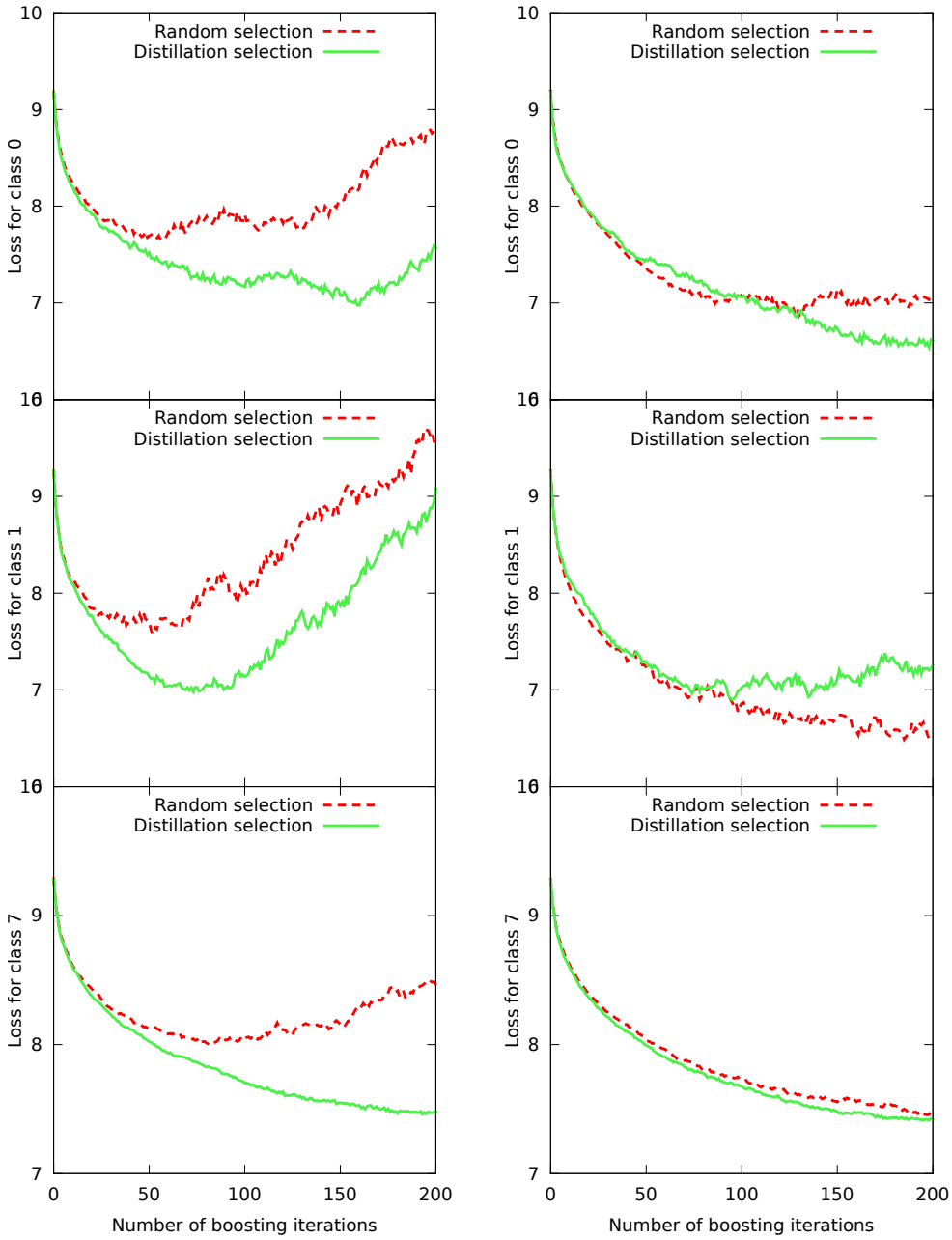


(a) With 1,000 images from MNIST



(b) With 5,000 images from MNIST

Figure 4: Analysis of the training set augmentated with distortions – The graph shows the test accuracy as a function of the number of distortions added per digit (5 distortions means that the second classifier was trained with $N+5N$ images). The initial test accuracy for 1,000 (resp. 5,000) is 94.47% (resp. 97.37%). First we see that adding distorted images helps. Second, we see that the GEEM procedure is able to pick "better" samples when few are added (from 1 to 7) but that random does better after that. See figure 5 for an explanation from the loss point of view.



(*a*) Adding 3 deformations per digit (*b*) Adding 7 deformations per digit

Figure 5: Analysis of the loss on validation samples for random and GEEM training samples – GEEM aims at discarding samples whose Boosting response can already be predicted with selected samples. Here we analyze the Boosting loss of some validation samples. These samples were discarded by the GEEM procedure. We see that their loss is even more reduced than training with a set of random distortions. Figure 4 show that adding more than 7 GEEM distortions does not help. This is what we can see on figure 5(*b*), class 1 where the loss is not as low for GEEM.

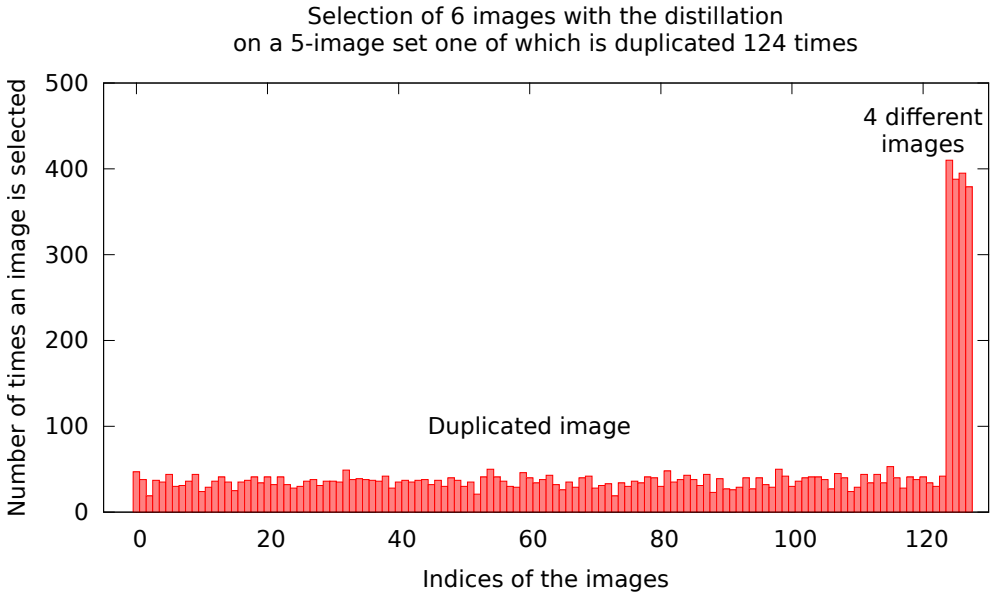


Figure 6: Analysis of the selection of images in a set containing duplicates – In this experiment, we build a set of 128 MNIST images of the same class from 5 different images and duplicate the first one 124 times. The set is thus 124 times the same image (indices from 1 to 124) alongside 4 different images (indices from 125 to 128). We run the distillation process to select 6 images out of the 128 with buckets of size 16. We run this experiment 1,000 times selecting randomly a class each time and count how many times the images where chosen. We see that the 4 unique images tend to be selected more often than the duplicates which shows that the GEEM procedure manages to build a diverse set of images.

5.4.2. How does GEEM handle distortions?

We analyze how GEEM behaves in comparison to randomness. First, we choose one image at random, generate one random distortion and compute the $\mathcal{L}_2$ norm between them. Second, we pick one image at random, generate 100 random distortions (one of which is the original) and use GEEM to discard all of them but one and compute the the $\mathcal{L}_2$ norm between the original and the remaining distortion. We repeat this experiment 5,000 times.

Histogram in Figure 7 shows the distribution of the $\mathcal{L}_2$ norm between the original and the distortion for both methods. We see that in comparison to the random set-up, the GEEM procedure will tend to choose a distortion different from the original which once again shows that GEEM can build a diverse set of samples given the initial classifier.

6. Conclusion

We presented a novel algorithm that extends a recent machine learning algorithm to make it applicable to large-scale data. The proposed extension not only allows the algorithm

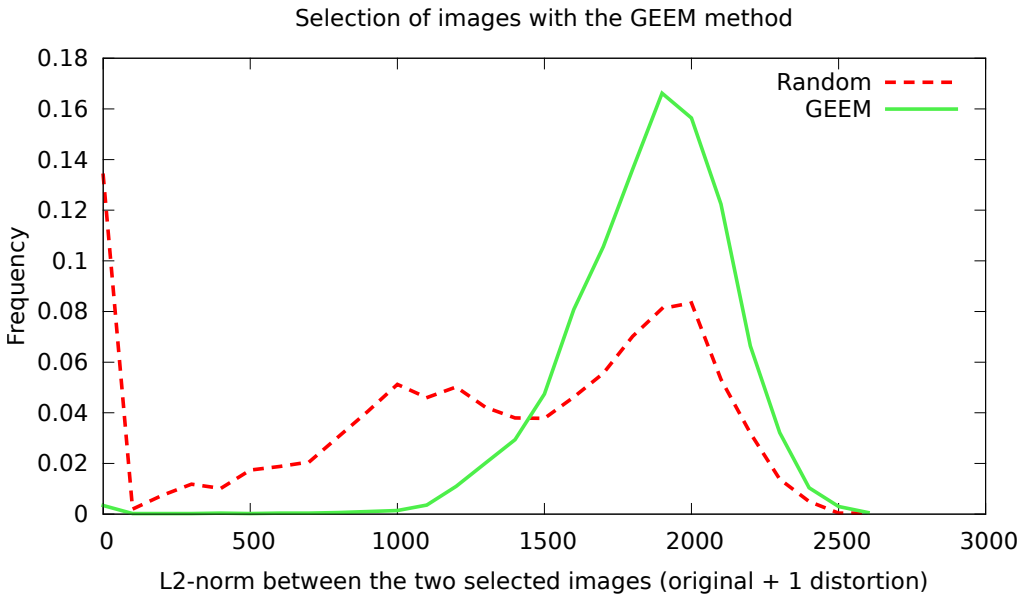


Figure 7: Analysis of the difference between selecting random distortions and distilled distortions to augment the training set – In this experiment, we compute the $\mathcal{L}_2$ norm between an original image and a distortion either selected randomly or through distilling it from a set of 100 distortions. We run the experiment 5,000 times and plot a histogram of the distances between the original image and the chosen distortion. The figure shows that the distillation process tends to select images which are different from the original leading to diversity. The peak observed for the random part is due to the fact that we do not interpolate the gray levels of the distorted image. When the rotation angle is too small, the resulting image is the same as the original. This does not occur for the distillation process because it manages not to select duplicates (see experiment of figure 6).

to scale-up but also allows for a computation/memory trade-off in its application due its ability to be processed in a parallel manner.

Through experimental results we show the relevancy of the method for computer vision by extending the state-of-the-art in two important applications, namely bootstrapping for object detection and dataset augmentation through distortions.

We furthermore analyzed the method not only from a theoretical perspective, i.e. its complexity, but also experimentally showing that the method not only performs well but in fact has a number of desirable properties that actually appear in practice.

In future work we plan to extend the distillation algorithm to work with a greater family of predictors, in particular with support vector machines. We are especially interested in the applicability of such approaches to the family of deformable part models.

Another line of current research is a more thorough theoretical analysis of jointly informative subset selection and in particular its relation to batch active learning.

Acknowledgments

Olivier Canévet was supported by the Swiss National Science Foundation under grant 200021-140912 – DASH, and Leonidas Leonidas was supported by the Hasler Foundation through the MASH2 project.

References

Joseph K. Bradley and Robert E. Schapire. Filterboost: Regression and classification on large datasets. In *NIPS*, 2007.

Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.

Dorin Comaniciu, Peter Meer, and Senior Member. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

Carlos Domingo and Osamu Watanabe. Madaboost: A modification of adaboost. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, COLT '00, pages 180–189, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-703-X. URL http://dl.acm.org/citation.cfm?id=648299.755176.

P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, Sept 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.167.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

Bharath Hariharan, Jitendra Malik, and Deva Ramanan. Discriminative decorrelation for clustering and classification. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part IV*, ECCV'12, pages 459–472, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33764-2. doi: 10.1007/978-3-642-33765-9_33. URL http://dx.doi.org/10.1007/978-3-642-33765-9_33.

J.F. Henriques, J. Carreira, R. Caseiro, and J. Batista. Beyond hard negative mining: Efficient detector learning via block-circulant decomposition. In *proceedings of the IEEE International Conference on Computer Vision*, 2013.

Balázs Kégl and Róbert Busa-Fekete. Boosting products of base classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 497–504. ACM, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

L. Lefakis and F. Fleuret. Reservoir boosting : Between online and offline ensemble learning. In *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, 2013.

X. Li, C. G. M. Snoek, M. Worring, D. C. Koelma, and A. W. M. Smeulders. Bootstrapping visual categorization with relevant negatives. *IEEE Transactions on Multimedia*, 15(4): 933–945, June 2013.

Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 03*, ICPR '06, pages 850–855, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2521-0. doi: 10.1109/ICPR.2006.479. URL http://dx.doi.org/10.1109/ICPR.2006.479.

Margarita Osadchy, Daniel Keren, and Bella Fadida-Specktor. Hybrid classifiers for object classification with a rich background. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part V*, ECCV'12, pages 284–297, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33714-7. doi: 10.1007/978-3-642-33715-4_21. URL http://dx.doi.org/10.1007/978-3-642-33715-4_21.

Daniel L. Ruderman and William Bialek. Statistics of natural images: Scaling in the woods. In *NIPS*, pages 551–558, 1993.

Jamie Shotton, Andrew Blake, and Roberto Cipolla. Contour-based learning for object detection. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01*, ICCV '05, pages 503–510, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2334-X-01. doi: 10.1109/ICCV.2005.63. URL http://dx.doi.org/10.1109/ICCV.2005.63.

B.V. Srinivasan and R. Duraiswami. Efficient subset selection via the kernelized Rényi distance. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1081–1088, Sept 2009. doi: 10.1109/ICCV.2009.5459395.

P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.