Learning sparse deep neural networks using efficient structured projections on convex constraints for green AI

Michel Barlaud [1], **Antonin Chambolle** [2] **and Frédéric Guyard** [3]

(1) Univ. Côte d'Azur & CNRS, Sophia Antipolis, (2) CEREMADE, CNRS & Paris-Dauphine PSL (3) Orange Lab Sophia Antipolis

Abstract

Deep neural networks (DNN) have been applied recently to different domains and perform better than classical state-of-the-art methods. However the high level of performances of DNNs is most often obtained with networks containing millions of parameters and for which training requires substantial computational power. To deal with this computational issue proximal regularization methods have been proposed in the literature but they are time consuming.

In this paper, we propose instead a constrained approach. We studied algorithms for different constraints: the classical ℓ_1 unstructured constraint and structured constraints such as the $\ell_{2,1}$ constraint (Group LASSO). We propose a new $\ell_{1,1}$ structured constraint for which we provide a new projection algorithm. Finally, we used the recent "Lottery optimizer" replacing the threshold by our $\ell_{1,1}$ projection. We demonstrate the effectiveness of this method with three popular datasets (MNIST, Fashion MNIST and CIFAR). Experiments with these datasets show that our projection method using this new $\ell_{1,1}$ structured constraint provides the best decrease in memory and computational power.

Classical Group LASSO structured constraint

Group LASSO was first introduced in [3]. The main idea is to enforce parameters of different classes to share common features. Group sparsity reduces so complexity by eliminating entire features. It consists in using the $\ell_{2,1}$ norm for the constraint on W, which is defined as follows. The rowwise $\ell_{2,1}$ norm of a $d \times k$ matrix W (whose rows are denoted w_i, $i = 1, \ldots, d$) is

$$\|W\|_{2,1} := \sum_{i=1}^{d} \|w_i\|_2. \tag{1}$$

A new $\ell_{1,1}$ structured constraint

Unfortunately, the Group LASSO structured constraint algorithm [1] does not provide efficient sparsity. Thus we propose the following algorithm.

Algorithm 1 Projection on the $\ell_{1,1}$ norm ($\text{proj}_{\ell_1}(V, \eta)$ is the projection on the ℓ_1-ball of radius η)

Input: V, η
$t := \text{proj}_{\ell_1}((\|v_i\|_1)_{i=1}^d, \eta)$
for $i = 1, \ldots, d$ **do**
  $w_i := \text{proj}_{\ell_1}(v_i, t_i)$
end for
Output: W

Analysis of the structured-$\ell_{1,1}$ projection: While the algorithm above does not strictly speaking define a projection, its solution corresponds to solving a sort of bi-level projection and is easily shown to be obtained as a limit, as ε goes to zero, of the minimizers of the convex problem:

$$\min_{\substack{\sum_i t_i \le \eta \\ \sum_j |w_{i,j}| - t_i \le 0}} \sum_{i=1}^{d} \left(\sum_{j=1}^{k} |v_{i,j}| - t_i \right)^2 + \varepsilon \sum_{i=1}^{d} \sum_{j=1}^{k} (v_{i,j} - w_{i,j})^2. \tag{P_ε}$$

In other words, $(t, w) \in \mathbb{R}_+^d \times \mathbb{R}^{d \times k}$ can be seen as the projection of $((\sum_j |v_{i,j}|)_{i=1}^d, v)$ onto the convex set $\{(t, w) : \sum_{i=1}^d t_i \le \eta, \sum_{j=1}^k |w_{i,j}| \le t_i \ \forall i = 1, \ldots, d\}$ in a degenerate (lexicographic) distance which infinitely favors the t component over the w variable.

Lottery optimizer

Following the work of Frankle and Carbin, who proposed an algorithm to find sparse sub-networks. We replaced their thresholding by our $\ell_{1,1}$ projection and devised the following algorithm:

Algorithm 2 Projection on the $\ell_{1,1}$ norm. Here $\text{proj}_{\ell_1}(V, \eta)$ is the projection on the ℓ_1-ball of radius η, $\nabla L(W, M_0)$ is the masked gradient with binary mask M_0, f is the ADAM optimizer and γ is the learning rate

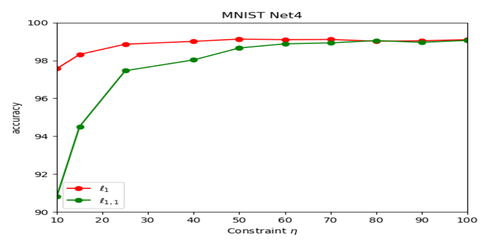
Input: $W*, \gamma, \eta$
for $n = 1, \ldots, N(epochs)$ **do**
  $V \leftarrow f(W, \gamma, \nabla L(W))$
end for
$t := \text{proj}_{\ell_1}((\|v_i\|_1)_{i=1}^d, \eta)$
for $i = 1, \ldots, d$ **do**
  $w_i := \text{proj}_{\ell_1}(v_i, t_i)$
end for
Output: W, M_0
Input: $W*$
for $n = 1, \ldots, N(epoch)$ **do**
  $W \leftarrow f(W, \gamma, \nabla L(W, M_0))$
end for
Output: W

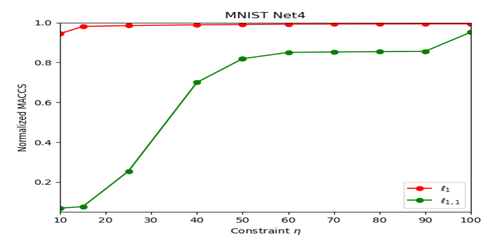
Results and comparison of methods:

Results on MNIST with a convolutionnal Network

We selected the popular MNIST dataset containing 28×28 grey-scale images of handwritten digits of 10 classes (from 0 to 9). This dataset consists of a training set of 60,000 instances and a test set of 10,000 instances.

We consider a neural network with two convolutional layers and two linear layers denoted as Net4.

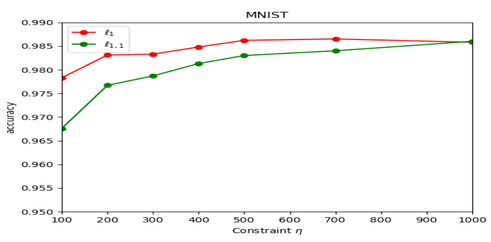


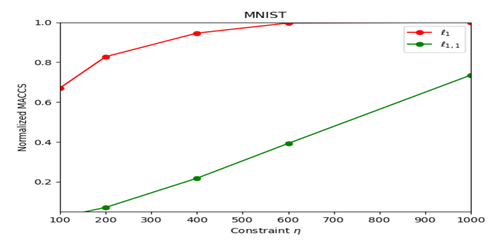


Methods	Memory (kBytes)	MACCs (k-MACCs)	Accuracy (%)
Adam	33.64	480	99.
Proj ℓ_1 ($\eta = 80$)	10.9	477	99.01
Proj $\ell_{1,1}$ ($\eta = 25$)	2.1	122	97.4

Results on MNIST with two Linear fully connected Networks

We used first a linear fully connected network (LFC4) with an input layer of d neurons, 4 hidden layers followed by a RELU activation function and a latent layer of dimension k. We used a linear fully connected network (LFC4) with an input layer of d neurons, 4 hidden layers followed by a RELU activation function and a latent layer of dimension k.





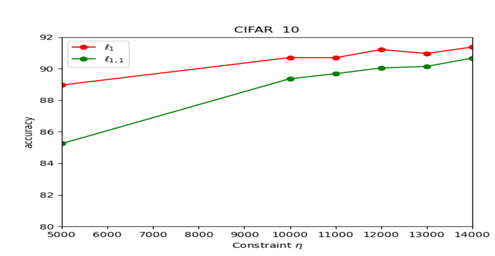
Second, we provide results using LeNet 300/100 a popular Linear Fully connected Network for benchmarking [2].

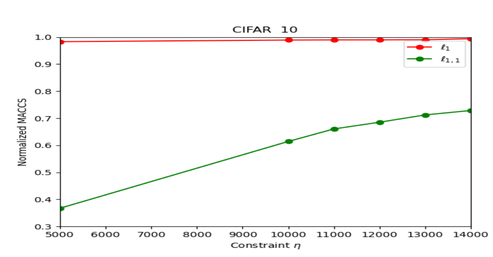
Methods	Memory (kBytes)	MACCs (k-MACCs)	Accuracy (%)
ADAM	477	266.2	98.21
Proj ℓ_1 ($\eta = 200$)	61	189	98.03
Proj $\ell_{1,1}$ ($\eta = 200$)	32	62	96.4
Proj $\ell_{1,1}$ ($\eta = 400$)	83	150	97.8
Proj $\ell_{2,1}$ ($\eta = 50$)	164	257	98.1
Tartaglione [2]	33.7	-	96.6

This table shows that our method outperforms the state-of-the-art [2] in terms of bytes accuracy compromise. The figures and tables show that the main advantage of our method using the $\ell_{1,1}$ constraint over ℓ_1 is the reduction of the calculation cost (MACCs) by a factor 14 when using a LFC4 network which is crucial for low capacity devices such as smartphones. The performance in MACCs using the $\ell_{2,1}$ constraint is intermediate between the use of ℓ_1 and $\ell_{1,1}$. Note that, to the best of our knowledge no results have been published in terms of FLOP reduction on this basis.

Results on CIFAR10

The CIFAR-10 data set is composed of 60,000 32x32 color images, 6,000 images per class, for a classification in 10 classes. The training set is made up of 50,000 images, while the remaining 10,000 are used for the testing set. We use *Simplenet*, a network composed of 13 blocks.





Methods	MACCs (M-MACCs)	Memory (M-Bytes)	Accuracy %
Adam	631.51	9.44	93.8
Proj ℓ_1 ($\eta = 13000$)	626.08	1.45	91.12
Proj $\ell_{1,1}$ ($\eta = 14000$)	441	0.86	91

The table and figures show a large global decrease in memory by a factor of 10. On the other hand the decrease in the calculation cost was about 30% for the $\ell_{1,1}$ constraint and almost null for ℓ_1 constraint.

References

[1] Michel Barlaud, Antonin Chambolle, and Jean-Baptiste Caillau. Robust supervised classification and feature selection using a primal-dual method. *arXiv cs.LG/1902.01600*, 2019.

[2] Enzo Tartaglione, Skjalg Lepsøy, Attilio Fiandrotti, and Gianluca Francini. Learning sparse neural networks via sensitivity-driven regularization. In *Advances in Neural Information Processing Systems*, pages 3878–3888, 2018.

[3] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.