Learning Network Size while Training with ShrinkNets

Guillaume Leclerc[1] Raul Castro Fernandez[1] Manasi Vartak[1] Martin Jaggi[2] Samuel Madden[1]

Abstract

Let's write the abstract at the end

1. Tentative outline

- We want to figure out what is the proper network size
- If we had a oversized network and for each neuron we would have an on/off switch, we could optimize the state of each switch to achieve any size/accuracy tradeof (we can prove that, do we care ?)
- Solving such problem is NP-Hard
- We could approximate the binary switch by relaxing them and doing L1 loss instead of L0 loss
- This is the definition of Shrinknets
- group sparsity tries to achieve a similar goal but with a different formulation
- How are they related ?
- If we add a specific constraint on our formulation we obtain the group sparsity one (proven)
- We conclude that without this constraint our formulation has more degree of freedom
- What happen when we drop this constraint
- The problem becomes non-convex and without a global minimum
- Not having a global minimum is bad, how can we solve that ?
- Adding an extra regularization parameter allow us to have a global minimum (a lot of them actually)
- Now we have a framework to switch on and of neurons
- We are still doing useless computation for neurons that will stay off forever, if we pruned them then we would speed up training
- Define the strategy to detect "forever dead" neurons
- (should we talk about the software architecture ?)
- now we evaluate the system
- First we show that our system has indeed more freedom than Group sparsity by showing that for any given amount of sparsity it can fit closer than the previous method
- On the same problem we show that removing neurons on the fly does not dramatically reduce accuracy (we might not be able to see any difference in performance for a linear/logistic regression though)
- We show that works with Bigger networks (multi layers perceptrons and CNNs), describe the training dynamics, discuss the shape we obtain
- Show that it is easier to do hyper-parameter optimization on the regularization parameter that the size of each layer. I think a nice experiment would be to plot the evolution of the final size of a two layer network (so going from a lambda to two sizes). And plot the trajectory in the "size space" as we decrease lambda. It would show how the system trades off the budget between the two layer in a nice and visual way
- Then we evaluate performance

[1]Computer Science and Artificial Intelligence Laboratories, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA [2]Swiss Federal Institute of Technology, Lausanne, Switzerland. Correspondence to: Guillaume Leclerc <leclerc@mit.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

2. Introduction

3. ShrinkNets

3.1. Motivation

3.2. Notations

In order to avoid any potential ambiguity, in this section we will describe in details the mathematical notations used in this article. Non-bold letters represent scalar values, while bold lowercase and upper case repectively denote vectors and matrices. $\boldsymbol{A}^T$ stands for the transpose of the matrix $\boldsymbol{A}$. Subscripts are used to index particular elements of vectors and matrices. $\boldsymbol{x}_i$, $\boldsymbol{A}_i$, $\left(\boldsymbol{A}^T\right)_j$ and $\boldsymbol{A}_{i,j}$ respectively correspond to the i^{th} component of $\boldsymbol{x}$, the i^{th} row of $\boldsymbol{A}$, the j^{th} column of $\boldsymbol{A}$ and the j^{th} component of the i^{th} row of $\boldsymbol{A}$. All the following definitions assume $\boldsymbol{A}$ to be an $n \times p$ matrix. For any vetor $\boldsymbol{b}$ with n components, we define $\operatorname{diag}(\boldsymbol{b})$ a $n \times n$ matrix such that: $\forall 1 \le i \le n$, $\operatorname{diag}(\boldsymbol{b})_{i,i} = \boldsymbol{b}_i$ and 0 otherwise. For any $l \in [0, +\infty]$ we define the norm: $\|\boldsymbol{A}\|_l = \left(\sum_{i=1}^n \sum_{j=1}^p |\boldsymbol{A}_{i,j}|^l\right)^{\frac{1}{p}}$. For the rest of this paper and unless stated otherwise, $\boldsymbol{y}$ will represent the output of a network, $\boldsymbol{x}$ the input, $\boldsymbol{b}$ a bias, λ regularization factors, Ω regularization methods, $\boldsymbol{\theta}$ general model parameters and a will stand for any element-wise activation function. The only constraint that we want to enforce is that $a(0) = 0$. We use $[\![u, v]\!]$ to denote inteveral of integers, $\boldsymbol{0}$ is the null vector (size depending on the context). $\#S$ is meant to represent the cardinality of a set S. To simplify the notation of function composition we use the following operator: $g(f(\boldsymbol{x})) = (f \circ g)(\boldsymbol{x})$ and for a long sequence of functions from f_1 to f_n we use: $f_n(...f_1(\boldsymbol{x})) = \left(\bigcirc_{k=1}^n f_k\right)(\boldsymbol{x})$.

3.3. Definition

$$\Omega_\lambda^s = \lambda \|\boldsymbol{\beta}\|_1 \tag{1}$$

4. Theoretical analysis

Our goal when designing ShrinkNets was to be able to remove inputs and outputs of layers. For classic fully connected layers, which are defined as :

$$f_{\boldsymbol{A},\boldsymbol{b}}(\boldsymbol{x}) = a(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}) \tag{2}$$

removing an input neuron j is equivalent to have $\left(\boldsymbol{A}^T\right)_j = \boldsymbol{0}$ and removing an output neuron i is the same as having $\boldsymbol{A}_i = \boldsymbol{0}$ and $\boldsymbol{b}_i = 0$. Solving optimization problems while trying to set entire groups of parameters to zero has been already studied and the most popular method is without doubt the group sparsity regularization [ref]. For any partitionning of the set of parameter defining a model in p groups: $\boldsymbol{\theta} = \bigcup_{i=1}^P \boldsymbol{\theta}_i$ we define it the following way:

$$\Omega_\lambda^{gp} = \lambda \sum_{i=1}^p \sqrt{\#\boldsymbol{\theta_i}}\, \|\boldsymbol{\theta_i}\|_2 \tag{3}$$

In the context of a fully-connected layer, the groups are either: columns of $\boldsymbol{A}$ if we want to remove inputs, or rows of $\boldsymbol{A}$ and the corresponding entry in $\boldsymbol{b}$ if we want to remove outputs. For simplicity, we will focus our analysis in the simple one-layer case. In this case filtering outputs does not make a lot of sense, this is why we will only consider the former case. The group sparsity regularization then becomes:

$$\Omega_\lambda^{gp} = \lambda \sum_{j=1}^p \left\| \left(\boldsymbol{A^T}\right)_{\boldsymbol{j}} \right\|_2 \tag{4}$$

Because $\forall i, \#\boldsymbol{\theta}_i = n$, To make the notation simpler, we now embed $\sqrt{n}$ inside λ.

Since group sparsity and ShrinkNets try to achieve the same goal we will try to understand their similarities and differences. First let's recall the two problems. The ShrinkNet problem is:

$$\min_{\boldsymbol{A},\boldsymbol{\beta}} \|\boldsymbol{y} - \boldsymbol{A}\operatorname{diag}(\boldsymbol{\beta})\,\boldsymbol{x}\|_2^2 + \Omega_\lambda^s \tag{5}$$

And the Group Sparsity problem is:

$$\min_{\boldsymbol{A}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \Omega_\lambda^{gp} \tag{6}$$

We can prove the under the condition: $\forall 1 \le j \le p, \left\| \left(\boldsymbol{A}^T\right)_j \right\|_2 = 1$ the two problems are equivalent (proposition A.1). However if we relax this constraint then shrinknet becomes non-convex and has no global minimum (propositions A.2 and A.3). Fortunately, by adding an extra term to the ShrinkNet regularization term we can proove that:

$$\min_{\boldsymbol{A},\boldsymbol{\beta}} \|\boldsymbol{y} - \boldsymbol{A}\operatorname{diag}(\boldsymbol{\beta})\,\boldsymbol{x}\|_2^2 + \Omega_\lambda^s + \lambda_2 \|A\|_p^p \tag{7}$$

has many global minimum (proposition A.4) for all $p > 0$. This is the reason we define the regularized ShrinkNet penalty:

$$\Omega_{\lambda,\lambda_2,p}^{rs} = \lambda \|\boldsymbol{\beta}\|_1 + \lambda_2 \|\boldsymbol{\theta}\|_p^p \tag{8}$$

In practice we observed that $p = 2$ or $p = 1$ are good choice, while the latter will also introduce additional sparsity in the parameters.

5. Speeding up training with pruning

6. Evaluation

6.1. Multi-Target Linear and Logistic regressions

6.2. Neuron Removal strategies

6.3. Convergence and training dynamics of Neural networks

6.4. Hyper-optimization of ShrinkNets

6.5. Transfer Learning with ShrinkNets

6.6. Performance

7. Related Work

8. Future Work

9. Conclusion

A. Appendix

A.1. Proofs of propositions

Unless specified, all the proofs consider the Multi-Target linear regression problem

Proposition A.1. $\forall (n,p) \in \mathbb{N}_+^2, \boldsymbol{y} \in \mathbb{R}^n, \boldsymbol{x} \in \mathbb{R}^p \lambda \in \mathbb{R}$

$$\begin{aligned}
&\min_{\mathbf{A}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \sum_{j=1}^{p} \left\| \left(A^T\right)_j \right\|_2 \\
=&\min_{\mathbf{A}', \boldsymbol{\beta}} \|\mathbf{y} - \mathbf{A}' \mathrm{diag}\left(\boldsymbol{\beta}\right) \mathbf{x}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 \\
&\text{s.t.} \forall j \in [\![1, p]\!], \left\| \left(A'^T\right)_j \right\|_2^2 = 1
\end{aligned}$$

Proof. In order to prove this statement we will show that for any solution $\boldsymbol{A}$ in the first problem, there exists a solution in the second with the exact same value, and vice-versa. We now assume we have a potential solution $\boldsymbol{A}$ for the first problem and we define $\boldsymbol{\beta}$ such that $\boldsymbol{\beta}_j = \left\| \left(\boldsymbol{A}^T\right)_j \right\|_2^2$, and $\boldsymbol{A}' = \boldsymbol{A} \left(\mathrm{diag}\left(\boldsymbol{\beta}\right)\right)^{-1}$. It is easy to see that the constraint on $\boldsymbol{A}'$ is statisfied by construction. Now:

$$\begin{aligned}
&\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \sum_{j=1}^{p} \left\| \left(A^T\right)_j \right\|_2 \\
=& \|\mathbf{y} - \mathbf{A}' \mathrm{diag}\left(\boldsymbol{\beta}\right) \mathbf{x}\|_2^2 + \lambda \sum_{j=1}^{p} \left\| \left(A'^T\right)_j \beta_j \right\|_2 \\
=& \|\mathbf{y} - \mathbf{A}' \mathrm{diag}\left(\boldsymbol{\beta}\right) \mathbf{x}\|_2^2 + \lambda \sum_{j=1}^{p} |\beta_j| \cdot 1 \\
=& \|\mathbf{y} - \mathbf{A}' \mathrm{diag}\left(\boldsymbol{\beta}\right) \mathbf{x}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1
\end{aligned}$$

Assuming we take an $\boldsymbol{A}'$ that satisfy the constraint and a $\boldsymbol{\beta}$, we can define $\boldsymbol{A} = \boldsymbol{A}' \mathrm{diag}\left(\boldsymbol{\beta}\right)$. We can apply the same operations in reverse order and obtain an instance of the first problem with the same value. We can now see that the two problems must have the same minimum otherwise we would be able to construct a solution to the other with exact same value. □

Proposition A.2.

$$\|\mathbf{y} - \mathbf{A} \mathrm{diag}\left(\boldsymbol{\beta}\right) \mathbf{x}\|_2^2$$

is not convex in $\boldsymbol{A}$ and $\boldsymbol{\beta}$.

Proof. To prove this we will take the simplest instance of the problem: with only scalars. We have $f(a, \beta) = (y - a\beta x)^2$. For simplicty let's take $y =$ and $x > 0$. If we take two candidates $s_1 = (0, 2)$ and $s_2 = (2, 0)$, we have $f(s_1) = f(s_2) = 0$. However $f(\frac{2}{2}, \frac{2}{2}) = x > \frac{1}{2} f(0, 2) + \frac{1}{2} f(2, 0)$, which break the convexity property. Since we showed that a particular case of the problem is non-convex then necessarily the general cannot be convex. □

Proposition A.3.

$$\min_{\mathbf{A}, \boldsymbol{\beta}} \|\mathbf{y} - \mathbf{A} \mathrm{diag}\left(\boldsymbol{\beta}\right) \mathbf{x}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1$$

has no solution if $\lambda > 0$.

Proof. Let's assume this problem has a minimum $\boldsymbol{A}^*, \boldsymbol{\beta}^*$. Let's consider $2\boldsymbol{A}^*, \frac{1}{2}\boldsymbol{\beta}^*$. Trivially the first component of the sum is identical for the two solutions, however $\lambda \left\| \frac{1}{2} \boldsymbol{\beta} \right\| < \lambda \|\boldsymbol{\beta}\|$. Therefore $\boldsymbol{A}^*, \boldsymbol{\beta}^*$ cannot be the minimum. We conclude that this problem has no solution. □

Proposition A.4. For this proposition we will not restrict ourselves to single layer but the composition of an an arbitrary large (n) layers as defined individually as $f_{\boldsymbol{A}_i, \boldsymbol{\beta}_i, \boldsymbol{b}_i}(x) = a(\boldsymbol{A_i} \mathrm{diag}\left(\boldsymbol{\beta_i}\right) \boldsymbol{x} + \boldsymbol{b_i})$. The entire network follows as: $N(\boldsymbol{x}) = \left(\bigcirc_{i=1}^{n} f_{\boldsymbol{A_i}, \boldsymbol{\beta_i}, \boldsymbol{b_i}}\right)(\boldsymbol{x})$. For $\lambda > 0$, $\lambda_2 > 0$ and $p > 0$ we have:

$$\min \|\boldsymbol{y} - N(\boldsymbol{x})\|_2^2 + \Omega^{rs}_{\lambda, \lambda_2, p}$$

has at least 2^k global minimum where $k = \sum_{i=1}^{n} \#\boldsymbol{\beta_i}$

Proof. First let's prove that there is at least one minimum to this problem. The two components of the expression are always positive so we know that this problem is bounded by below by 0. Let's assume this function does not have a minimum. Then there is a sequence of parameters $(S_n)_{n>0}$ such that the function evaluated at that point convereges to the infimum of the problem. Since the function is defined everywhere does not have a minimum then this sequence must diverge. Since the entire sequence deverge the there is at least one individual parameter that diverges. First case, the parameter is a component k of some $\boldsymbol{\beta_i}$ for some i. Necessarly $\|\boldsymbol{\beta_i}\|_1$ diverge towards $+\infty$, which is incompatible with the fact that (S_n) converges to the infimum. We can have the exact same argument if the diverging parameter is in $\boldsymbol{A_i}$ or $\boldsymbol{b_i}$ because $p > 0$. Since there is always a contradiction then our assumption that the function has no global minimum must be false. Therefore, this problem has at least one global minimum.

Let's consider one optimal solution of the problem. For each component k of $\boldsymbol{\beta_i}$ for some i. Negating it and negating the k^{th} column of $\boldsymbol{A_i}$ does not change the the first part of the objetive because the two factors cancel each other. The two norms do not change either because by definition the norm is independant of the sign. As a result these two sets of parameter have the same value and are both global minimum. It is easy to see that going from this global minimum we can decide to negate or not each element in each $\boldsymbol{\beta_i}$. We have a binary choice for each parameter, there are $k = \sum_{i=1}^{n} \#\boldsymbol{\beta_i}$ parameters, so we have at least 2^k global minima.

$\square$

A.2. Experiment details

A.2.1. Multi-Target Linear and Logistic regressions

A.2.2. Neuron Removal strategies

A.2.3. Convergence and Training Dynamics of Neural networks

A.2.4. Hyper-optimization of ShrinkNets

A.2.5. Transfer Learning with ShrinkNets

A.2.6. Performance