Playing the Lottery With Concave Regularizers for Sparse Trainable Neural Networks

(Article begins on next page)

Playing the Lottery With Concave Regularizers for Sparse Trainable Neural Networks

Giulia Fracastoro, *Member, IEEE*, Sophie M. Fosson, *Member, IEEE*, Andrea Migliorati, and Giuseppe C. Calafiore, *Fellow, IEEE*

***Abstract*— The design of sparse neural networks, i.e., of networks with a reduced number of parameters, has been attracting increasing research attention in the last few years. The use of sparse models may significantly reduce the computational and storage footprint in the inference phase. In this context, the lottery ticket hypothesis (LTH) constitutes a breakthrough result, that addresses not only the performance of the inference phase, but also of the training phase. It states that it is possible to extract effective sparse subnetworks, called *winning tickets*, that can be trained in isolation. The development of effective methods to *play the lottery*, i.e., to find winning tickets, is still an open problem. In this article, we propose a novel class of methods to play the lottery. The key point is the use of concave regularization to promote the sparsity of a relaxed binary mask, which represents the network topology. We theoretically analyze the effectiveness of the proposed method in the convex framework. Then, we propose extended numerical tests on various datasets and architectures, that show that the proposed method can improve the performance of state-of-the-art algorithms.**

***Index Terms*— Concave regularization, lottery ticket hypothesis (LTH), neural network pruning, sparse optimization.**

I. Introduction

NEURAL network pruning refers to the sparsification of a neural architecture by removing unnecessary parameters, i.e., either connections (weights) or neurons. As a matter of fact, neural networks are often overparametrized and their size can be significantly decreased while keeping a satisfactory accuracy. Pruning allows us to reduce the computational costs, storage requirements, and energy consumption of a neural network in the inference phase; see, e.g., [1], [2], [3], [4]. This has several advantages, ranging from the circumvention of overfitting [1] to the possibility of implementing deep learning in embedded mobile applications [5], [6].

Learning pruned (or sparse) neural networks is a challenging task, which has drawn substantial attention in the last years. In the literature, two main approaches are considered. The classic one is *dense-to-sparse*: the model is dense at the beginning and during the training, while the output of the training is a sparse network. In this context, a common practice is the following three-stage iterative procedure [2], [7]: first, the dense method is trained, then the sparse subnetwork is extracted, and finally the subnetwork is retrained, by starting from the weights of the trained dense model; this last stage is known as fine-tuning (FT). Most of the proposed dense-to-sparse training methods reduce the number of nonzero parameters (i.e., the ℓ_0 norm of the parameter vector) by pruning the weights with the largest magnitude [2], [8] or via ℓ_0 regularization [3]. More recently, ℓ_1 and concave regularizations have been studied and tackled through proximal gradient methods [9], [10]. The benefit of the dense-to-sparse approach is the acceleration of the inference task by using the sparse model; nevertheless, the training remains computationally intense.

The second and more recent approach, known as *sparse-to-sparse*, tackles the computational burden of the training phase. Basically, it states that we can learn the sparse architecture and then train it in isolation. However, learning a sparse topology that can be trained in isolation is very challenging. In particular, it has been observed that retraining a sparse network obtained through a dense-to-sparse method from a random initialization often yields a substantial loss of accuracy. A way to circumvent this problem is *rewinding* the weights to the original initialization of the dense model, as proposed in [4]. More precisely, in [4], the authors conjecture the lottery ticket hypothesis (LTH): a dense, randomly initialized neural network N_d contains a small subnetwork N_s that achieves a test accuracy comparable to the one of N_d, with a similar number of iterations, provided that N_s is trained in isolation with the same initialization of N_d.

The effective subnetworks mentioned in the LTH are named *winning tickets* as they have won the *initialization lottery*. While their existence is proven in [4], their extraction is not straightforward. As a matter of fact, the development of effective algorithms to win the lottery is still an open problem. In [4], a method based on iterative magnitude pruning (IMP) is proposed, which still represents the state-of-the-art heuristic to play the lottery. In [11], IMP is refined to deal with instability: the weights are rewinded to an early iteration $k > 0$ instead of the initial θ_0 [11, Sec. III]. Thus, subnetworks are no longer randomly initialized, and they are denoted as matching tickets [11, Sec. IV]. In [12], an ℓ_0 regularization approach is proposed to search winning tickets, based on continuous

Manuscript received 25 November 2022; revised 29 December 2023; accepted 23 February 2024. The work of Sophie M. Fosson was supported by the Project Nord-Ovest Digitale e Sostenibile (NODES) which has received funding from the Ministry of University and Research (MUR)–M4C2 1.5 of National Recovery and Resilience Plan (PNRR) under Grant ECS00000036. The work of Andrea Migliorati was supported by the European Union under Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, Partnership on “Telecommunications of the Future” (PE00000001–Program “RESTART”). *(Corresponding author: Giulia Fracastoro.)*

Giulia Fracastoro, Andrea Migliorati, and Giuseppe C. Calafiore are with the Department of Electronics and Telecommunications (DET), Politecnico di Torino, 10129 Turin, Italy (e-mail: giulia.fracastoro@polito.it).

Sophie M. Fosson is with the Department of Control and Computer Engineering (DAUIN), Politecnico di Torino, 10129 Turin, Italy.

Digital Object Identifier 10.1109/TNNLS.2024.3373609

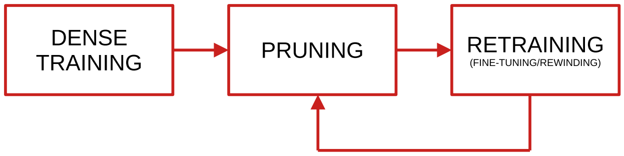

Fig. 1. Three-stage pipeline for pruning. The initialization of the retraining stage is done either with FT or rewinding.

sparsification, which outperforms the accuracy of IMP in some numerical experiments. Nevertheless, such enhancement is obtained at the price of repeated rounds, which causes a slower sequential search, as discussed in [12, Sec. V.1].

In this work, we propose a novel method to play the lottery, i.e., to find sparse neural networks that can be trained in isolation by rewinding. The key steps of our approach are the following: we define a binary relaxed mask, which describes the network topology, and we optimize it by applying a continuous, concave regularization. In particular, we consider ℓ_1 (which is also convex) and logarithmic regularizers. The rationale behind continuous regularization is that most of the sparse-to-sparse methods used to play the lottery [4], [12] are ℓ_0-based, which yields hard decisions. For example, in IMP, the percentage of surviving parameters is priorly set, possibly causing the removal of important weights. Instead, a continuous regularization may yield softer decisions in the pruning process, and, as a consequence, more accurate models. To the best of the authors' knowledge, this is the first sparse-to-sparse method employing an ℓ_1/concave setting.

As discussed throughout the article, the use of a relaxed binary mask allows us to train the model via projected gradient descent even though the proposed sparsity-promoting regularization is nondifferentiable. On the other hand, we show that strictly concave regularization is particularly efficient when applied on binary variables.

The main contributions of this article can be summarized as follows. We develop a novel strategy to play the lottery that leverages the use of sparsity-promoting regularization and relaxed binary masks. Then, we provide theoretical conditions under which our approach guarantees to find an optimal sparse mask, at least in the case of convex loss functions. Finally, we extensively validate the proposed method on various datasets and architectures. The article is organized as follows. In Section II, we illustrate the background and previous work on neural network pruning and LTH, to contextualize the proposed contribution. In Section III, we present the proposed method, which is theoretically analyzed in Section IV. Section V is devoted to experiments, comparisons, and discussions. Finally, we draw some conclusion.

II. Related Work

In recent years, the literature on neural network pruning and LTH has significantly grown, and a complete overview is beyond our purposes. In this section, we review the main works that, for different motivations, are related to our approach. In Table I, we report a list of relevant works and their properties.

A. Dense-to-Sparse Methods

Regarding the dense-to-sparse approach, several approaches include a regularization term in the loss function to sparsify the model, the most popular being the ℓ_0-norm [3], ℓ_1-norm [9] and concave ℓ_p norms [10]. In these works, training and sparsification are performed in a joint optimization problem. In contrast, in the method proposed in [2], training and pruning are separated tasks, and the pruned architecture is iteratively retrained, according to the three-stage pipeline depicted in Fig. 1. The retraining stage is initialized with FT, i.e., by starting from the parameters obtained in the previous training stage. Such a three-stage procedure with FT is very common in the dense-to-sparse approach, see, e.g., [2], [6], [13], [14].

Among more recent works, in [15] a different regularization is proposed, based on the neural sensitivity, to learn sparse topologies with a structure. In [16], an energy-based pruning method is developed, combined with a dropout approach. In [17], [18], [19], [20], and [21], specific methods for filter/channel pruning in convolutional neural networks are developed.

B. Sparse-to-Sparse Methods

The above-mentioned three-stage pipeline is also popular in the sparse-to-sparse approach [4], [12], but with rewinding: in the retraining stage, the parameters are reinitialized by rewinding them to the original initialization; see Section I. In the IMP method [4], which is by far the most common sparse-to-sparse approach, pruning is performed by removing the $p\%$ parameters with the smallest magnitude. This approach has some drawbacks. As p is priorly set, this may result either in an excessive sparsification with the removal of important weights or in an insufficient sparsification that retains superfluous parameters. Moreover, very similar weights may be either retained or removed to fulfill $p\%$, which contradicts the principle of saving the most relevant parameters. In [12], the hard effect of magnitude pruning is mitigated by a continuous relaxation of the ℓ_0 regularization. However, as discussed in Section I, in practice this does not outperform IMP.

In this article, we tackle these issues by introducing ℓ_1/log regularizers to induce sparsity. This results in a softer approach where the output of the optimization/training stage is not expected to be an exact binary mask, but a relaxed mask that can be used for pruning the dense network by setting a suitable threshold $\alpha \in [0, 1]$. Also, the dense-to-sparse approach presented in [2] proposes to set a threshold instead of removing a fixed percentage of the weights. However, in [2], the threshold is set on the weights of the parameters, which basically can assume any real value and whose range may vary at each layer of the network. Therefore, setting such a threshold may be critical and it requires some prior knowledge of the range of the parameters. In addition, defining a unique threshold for the entire network can be troublesome because weights from different layers might have different orders of magnitude.

TABLE I

CLASSIFICATION OF MAIN WORKS ON NEURAL NETWORK PRUNING. NOTATION: D2S = DENSE-TO-SPARSE APPROACH; S2S = SPARSE-TO-SPARSE APPROACH; 3S = METHODS EXPLOITING A THREE-STAGE ITERATIVE PROCEDURE, AS IN FIG. 1. 3S METHODS MAY REINITIALIZED EITHER WITH FT OR REWINDING (RW)

Article - Approach	D2S	S2S	3S
[2] - Magnitude pruning	✓		✓ FT
[3] - ℓ_0 regularization	✓		
[9] - ℓ_1 regularization	✓		
[10] - ℓ_p regularization, $p \in [0, 1]$	✓		
[15] - sensitivity regularization	✓		
[16] - energy-based dropout	✓		
[4] - IMP with rewinding		✓	✓ RW
[12] - ℓ_0 regularization		✓	✓ RW
This work - ℓ_1/log regularization		✓	✓ RW

Algorithm 1 Soft Mask Pruning With Concave Regularization

Input: $\theta_0 \in \mathbb{R}^d$, $m_0 = \frac{1}{2}(1, 1, \ldots, 1) \in \mathbb{R}^d$, $\alpha \in (0, 1)$, T

1: **for all** $t = 1, \ldots, T$ **do**

2: Initialization:

$\theta = \theta_0$, $m = m_{t-1}$

3: Optimization:

$\min_{\theta\in\mathbb{R}^d, m\in[0,1]^d} \mathcal{L}(x; m \odot \theta) + \lambda \mathtt{R}(m) \quad \rightarrow \quad (\theta_t, m_t)$

4: Pruning:

for each $i = 1, \ldots, d$, if $m_{t,i} < \alpha$, then $m_{t,i} = 0$

5: **end for**

III. PROPOSED METHOD

In this section, we present the proposed method.

Let us consider a neural network $f(x; \theta)$, where x represents the input data and $\theta \in \mathbb{R}^d$ are the weights. Let $\odot$ be the componentwise product between vectors. According to [4], playing the lottery consists of searching a binary mask $m \in \{0, 1\}^d$ with $\|m\|_0 \ll d$ such that $f(x; m \odot \theta)$ is a winning ticket, i.e., it achieves performance comparable to $f(x; \theta)$ when trained in isolation.

The seminal IMP search algorithm proposed in [4] consists of an iterative three-stage pipeline, as illustrated in Fig. 1. The retraining phase is performed by rewinding the parameters to $m_t \odot \theta_0$, where θ_0 is the original initialization and m_t is the current estimated mask.

The proposed method is as follows. We consider the mask m as an optimization variable and we use concave regularization to promote its sparsity. To avoid mixed-integer problems, we relax the binary mask and we consider it as a continuous variable

$$m \in [0, 1]^d. \tag{1}$$

Then, we jointly train and sparsify the network by solving

$$\min_{\theta\in\mathbb{R}^d, m\in[0,1]^d} \mathcal{L}(x; m \odot \theta) + \lambda \mathtt{R}(m) \tag{2}$$

where $\mathtt{R}$ is a sparsity-inducing, concave regularizer. This approach returns both a sparse relaxed mask and the trained weights. The relaxed mask in $[0, 1]^d$ can be interpreted as a relevance score or soft decision for each parameter. To refine these soft results, we prune the parameters with mask value below a small threshold $\alpha \in (0, 1)$, $\alpha \ll 1$. We remark that this is not a harsh pruning, which would contradict the proposed soft approach; instead, it is just a way to remove small weights expected to converge to zero. Furthermore, we notice that setting a threshold in $(0, 1)$ is more straightforward than setting a magnitude threshold on the parameters' values as in [2], because it does not require information about the parameters' range.

Then, we iterate the procedure as in Fig. 1, with a rewinding strategy, i.e., the three-stage external loop is analogous to the one of IMP.

The thorough procedure is summarized in Algorithm 1.

To complete the illustration of the algorithm, we introduce the family of considered regularizers $\mathtt{R}(m)$.

A. Sparsity-Inducing Concave Regularizers

The rationale behind concave regularization is as follows. While sparsity is represented by the ℓ_0-norm, its use as a regularizer is critical due to noncontinuity and nonconvexity. For this motivation, the ℓ_1-norm is often used, as it is the best convex approximation of the ℓ_0-norm, see [22]. In the presence of a convex cost function, by using the ℓ_1-norm, we recast the overall problem into convex optimization. A typical example is Lasso [22].

On the other hand, continuous concave regularizers have been observed to be more effective than ℓ_1 regularization, even though they introduce nonconvexity in the problem, as their shape is closer to the ℓ_0-norm.

In this work, we consider concave regularizers $\mathtt{R}(m)$ with the following property.

Assumption 1: $\mathtt{R}(m)$ is any function

$$\mathtt{R}(m) = \sum_{i=1}^{d} \mathtt{r}(m_i), \quad m_i \in [0, 1] \tag{3}$$

such that $\mathtt{r} : [0, 1] \rightarrow [0, 1]$ is continuous and differentiable in $(0, 1)$, concave, nondecreasing, and its image is $[0, 1]$.

In the literature of signal processing and sparse optimization, the most popular regularizers satisfying Assumption 1 are

1) *ℓ_1:* $\mathtt{r}_1(m_i) = m_i$, see [9], [10], [22].
2) *log:* $\mathtt{r}_\epsilon(m_i) = (\log((m_i + \epsilon)/\epsilon))/(\log((1 + \epsilon)/\epsilon))$ for any $\epsilon > 0$, see [23].

Other possible choices are ℓ_q norm, see [10], [24], and minimax concave penalties, see [25]. In this work, we focus our attention on ℓ_1 and log regularizers.

In the literature, the use of strictly concave regularization has arisen in the context of linear regression and compressed sensing, see, e.g., [23], [24], [26], [27], [28]. Then, it has been extended to several machine learning models; we refer the interested reader to the survey [29]. When the cost function is strictly convex, adding a strictly concave regularizer may keep the problem in the convex optimization framework, see, e.g., [30]. In contrast, the problem is more challenging when the cost function is nonconvex. This case is theoretically analyzed, e.g., in [31], [32], and [33], where much attention is devoted to proving the convergence of the applied algorithms (proximal methods and alternating direction method of multipliers).

Within the family of nonconvex cost functions, the case of neural networks is even more difficult. In fact, in deep learning, gradient-based algorithms are commonly used for training, which is in conflict with nondifferentiable regularization as in Assumption 1, as discussed in Section III-B.

B. Discussion on the Training Algorithm

The training phase of the proposed approach requires locally solving (2). In principle, the ℓ_1/ log regularization is critical for the application of gradient-based training algorithms, due the nondifferentiability in zero. However, in our approach, we regularize $m \in [0,1]^d$; thus, we can use a projected gradient-based algorithm, without differentiability issues.

We remark that ℓ_1 regularization is popular in deep learning and it is usually implemented in libraries such as TensorFlow and PyTorch. However, since ℓ_1 norm is not differentiable at zero, it is usually implemented with a subgradient approach, namely the gradient of $|x|$ is defined as $\text{sign}(x)$ for $x \neq 0$, and 0 for $x = 0$. This workaround may be effective in controlling the energy of the parameters, but subgradient iterates do not attain zero, thus sparsification fails. Moreover, subgradient methods are substantially slow and may result in oscillations. These drawbacks are illustrated in the numerical example in Section IV-C.

Recently, proximal operator methods are used instead of subgradient, see, e.g., [9], [10]. However, their application and convergence proof are critical and limited to some specific cases.

Given these considerations, the proposed relaxed binary mask turns out to be an effective alternative strategy to match the use of nondifferentiable, sparsity-promoting regularizers with standard gradient-based training algorithms.

IV. Theoretical Analysis

In this section, we prove theoretical results that support the effectiveness of the proposed method, by providing conditions that guarantee to extract an optimal sparse subnetwork topology. In particular, these results explain why a log regularizer may be preferable to ℓ_1. Finally, we show an illustrative example that corroborates the theoretical findings.

As a thorough analysis is quite complex, we focus on the following problem: we assume that a vector θ of trained parameters is available and that a sparse mask $\widetilde{m} \in \{0,1\}^d$ exists, such that $\widetilde{m} \odot \theta$ is a sparse topology with no substantial performance loss. Our aim is to estimate this optimal $\widetilde{m}$. To this purpose, we solve

$$\min_{m\in[0,1]^d} \mathcal{L}(x; m \odot \theta) + \lambda \mathrm{R}(m) \tag{4}$$

where $\mathrm{R}(m)$ satisfies Assumption 1.

On the one hand, this problem is addressed in the context of the strong LTH, where subnetworks are extracted from randomly weighted neural networks without modifying the weight values, see [34], [35]. On the other hand, if in (2) we proceed by alternated minimization over θ and m, (4) can be interpreted as a subproblem of (2).

Since, in this section, we consider m as the unique variable, for simplicity we write $\mathcal{L}(x; m \odot \theta) = \mathcal{L}(m)$. Let

$$m^\star = \operatorname*{argmin}_{m\in[0,1]^d} \mathcal{L}(m) + \lambda \mathrm{R}(m). \tag{5}$$

Then, $m^\star$ is an estimate of $\widetilde{m}$. To evaluate the accuracy of this estimate, we analyze the distance between $m^\star$ and $\widetilde{m}$. To this end, we assume that

$$\|\nabla \mathcal{L}(\widetilde{m})\|_\infty \leq \lambda. \tag{6}$$

This is always true by properly choosing λ. On the other hand, we expect $\|\nabla \mathcal{L}(\widetilde{m})\|_\infty$ to be small as far as $\widetilde{m}$ is close to a stationary point, i.e., if the model is amenable to effective sparsification.

To keep the analysis straightforward, we do the following convexity assumption.

Assumption 2: There exists $\gamma > 0$ such that

$$\mathcal{L}(m^\star) \geq \mathcal{L}(\widetilde{m}) + \nabla \mathcal{L}(\widetilde{m})^T (m^\star - \widetilde{m}) + \frac{\gamma}{2}\|m^\star - \widetilde{m}\|_2^2. \tag{7}$$

In other terms, we require strong convexity at least with respect to the points $\widetilde{m}$ and $m^\star$. Clearly, this is fulfilled if the loss is globally strongly convex.

A. Accuracy Analysis for ℓ_1 Regularization

In this section, we study the accuracy, in terms of distance between $\widetilde{m}$ and $m^\star$, in the case of ℓ_1 regularization. Specifically, we prove the following result.

Theorem 1: Let $h = m^\star - \widetilde{m}$. If $\mathrm{R}(m) = \|m\|_1$, then

$$\|h\|_2 \leq \frac{4\lambda\sqrt{k}}{\gamma}. \tag{8}$$

Proof: By definition of $m^\star$ in (5), we have

$$\mathcal{L}(m^\star) + \lambda \mathrm{R}(m^\star) \leq \mathcal{L}(\widetilde{m}) + \lambda \mathrm{R}(\widetilde{m}). \tag{9}$$

Let S and S_c denote the support of $\widetilde{m}$ and its complementary set, respectively. Moreover, h_S (respectively, h_{S_c}) is the subvector of h with components indexed in S (respectively, in S_c). Then

$$\begin{aligned}
\mathcal{L}(m^\star) - \mathcal{L}(\widetilde{m}) &\leq \lambda\|\widetilde{m}\|_1 - \lambda\|m^\star\|_1 \\
&= \lambda[\|\widetilde{m}\|_1 - \|\widetilde{m} + h\|_1] \\
&= \lambda\big[\|\widetilde{m}\|_1 - \|\widetilde{m}_S + h_S\|_1 - \|h_{S_c}\|_1\big] \\
&\leq \lambda\big[\|\widetilde{m}\|_1 - \|\widetilde{m}_S\| + \|h_S\|_1 - \|h_{S_c}\|_1\big] \\
&= \lambda\|h_S\|_1 - \lambda\|h_{S_c}\|_1.
\end{aligned} \tag{10}$$

Moreover,

$$\begin{aligned}
\nabla \mathcal{L}(\widetilde{m})^T (m^\star - \widetilde{m}) &\leq \|\nabla \mathcal{L}(\widetilde{m})\|_\infty^T \|h\|_1 \\
&\leq \lambda\|h\|_1.
\end{aligned} \tag{11}$$

Now, from (7) and (10), we have a lower bound and an upper bound for $\mathcal{L}(m^\star) - \mathcal{L}(\widetilde{m})$. By using also (11), we obtain

$$-\lambda\|h\|_1 + \frac{\gamma}{2}\|h\|_2^2 \leq \mathcal{L}(m^\star) - \mathcal{L}(\widetilde{m}) \leq \lambda\|h_S\|_1 - \lambda\|h_{S_c}\|_1. \tag{12}$$

Then,

$$\begin{aligned}\frac{\gamma}{2}\|h\|_2^2 &\le \lambda\|h\|_1 + \lambda\|h_S\|_1 - \lambda\|h_{S_c}\|_1 \\ &\le \lambda\|h_S\|_1 + \lambda\|h_{S_c}\|_1 + \lambda\|h_S\|_1 - \lambda\|h_{S_c}\|_1 \\ &\le 2\lambda\|h_S\|_1 \\ &\le 2\lambda\sqrt{k}\|h\|_2. \end{aligned} \tag{13}$$

Then, by dividing by $\|h\|_2$, we obtain the claim. ■

The statement of Theorem 1 provides an error bound on the estimate $m^\star$ of the optimal mask $\widetilde{m}$. This error bound is proportional to λ, which can be chosen as equal to $\|\nabla\mathcal{L}(\widetilde{m})\|_\infty$ based on (6). In other terms, the more the masked solution $\widetilde{m} \odot \theta$ is close to the nonmasked optimum θ, the more by solving the ℓ_1-regularized problem we obtain a good estimate. Similarly, the more sparsity k of $\widetilde{m}$ is small, the more $m^\star$ is a good estimate. The third parameter present in the error bound is γ, which is a measure of the convexity of the loss in Assumption 2. Then, the error bound is smaller if the loss is more convex, i.e., γ is larger. An enhancement of Theorem 1 can be obtained by replacing ℓ_1 with a more general strictly concave regularization.

B. Accuracy Analysis for Strictly Concave Regularization

In this section, we analyze the accuracy in case of strictly concave regularization.

Theorem 2: Let $h = m^\star - \widetilde{m}$. Let $\mathrm{R}(m)$ be strictly concave in $[0, 1]$ as in Assumption 1. Then

$$\|h\|_2 \le \frac{4\lambda\sqrt{k}}{\gamma} - \phi(m^\star) \tag{14}$$

where $\phi(m^\star) > 0$ is assessed in the proof.

Proof:

$$\mathcal{L}(m^\star) - \mathcal{L}(\widetilde{m}) \le \lambda\mathrm{R}(\widetilde{m}) - \lambda\mathrm{R}(m^\star). \tag{15}$$

Now, since $\widetilde{m} \in \{0, 1\}^d$, then $\mathrm{R}(\widetilde{m}) = \|\widetilde{m}\|_1 = \sum_{\widetilde{m}_i}$. Then,

$$\begin{aligned}\mathrm{R}(\widetilde{m}) - \mathrm{R}(m^\star) &= \|\widetilde{m}\|_1 - \mathrm{R}(m^\star) \\ &= \|\widetilde{m}\|_1 - \mathrm{R}(m^\star) \pm \|m^\star\|_1 \\ &\le \|h_S\|_1 - \|h_{S_c}\|_1 - \phi(m^\star) \end{aligned} \tag{16}$$

where

$$\phi(m^\star) = \mathrm{R}(m^\star) - \|m^\star\|_1 \ge 0. \tag{17}$$

■

Theorem 2 states that by replacing ℓ_1 with a strictly concave regularization, such as the logarithmic one, we obtain an error bound which is smaller than the one in Theorem 1 of $\phi(m^\star)$.

Let us discuss more in detail the role of $\phi(m^\star)$. We notice that if $m^\star \in \{0, 1\}^d$, then $\mathrm{R}(m^\star) = \|m^\star\|_1$, $\phi(m^\star) = 0$, and no enhancement is obtained by using a strictly concave regularizer. However, we know from the statements of Theorems 1 and 2 that $\|m^\star - \widetilde{m}\|_2 \le (4\sqrt{k}/\gamma)\lambda$. Then,

Lemma 1: If $(4\sqrt{k}/\gamma)\lambda < 1$ and $m^\star \in \{0, 1\}^d$, then $m^\star = \widetilde{m}$.

Therefore, if $(4\sqrt{k}/\gamma)\lambda < 1$, the unique feasible binary solution is $\widetilde{m}$. Moreover, if $m^\star \notin \{0, 1\}^d$, then $\phi(m^\star) > 0$ and we obtain a smaller error bound by using a strictly concave regularizer.

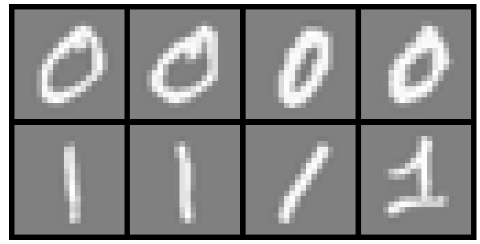

Fig. 2. Eight samples from the MNIST dataset.

C. Illustrative Example

To conclude the analysis, we show an illustrative example of Problem (4) with Assumption 2. Specifically, we consider a problem of binary classification performed through logistic regression (LG). We use the MNIST dataset [36] restricted to the digits 0 and 1. Each image is composed of $d = 400$ pixels. We consider 200 samples for each digit, 160 for training, and $N = 40$ for validation test. The corresponding cross-entropy loss function is

$$\mathcal{L}(X, y; \theta) = \sum_{i=1}^{N} y_i \log(\hat{y}_i) - (1 - y_i)\log(1 - \hat{y}_i)$$

where $y \in \{0, 1\}^N$ and $\hat{y} \in [0, 1]^N$, respectively, are the vectors of correct and estimated labels. More precisely,

$$\hat{y}_i = \frac{1}{1 + e^{X_i\theta}}$$

where $\theta \in \mathbb{R}^d$ is the vector of estimated parameters and $X \in \mathbb{R}^{N,401}$ is the validation dataset (including the intercept). The loss is convex; if an ℓ_2 regularization is added, we have strong convexity as in Assumption 2.

As we can see in Fig. 2, many pixels (e.g., the ones toward the image borders) are not significant for classification, therefore there is room for sparsification. Then, we tested different approaches for sparsification. First, we considered a classic ℓ_1 regularization, i.e., we minimize $\mathcal{L}(X, y; \theta) + \lambda\|\theta\|_1$. Second, we applied ℓ_1 and logarithmic regularizations to the mask. More precisely, given a vector $\hat{\theta}$ of parameters, e.g., obtained via LG, we aim at finding a binary mask $m \in \{0, 1\}^d$ that selects the significant pixels, i.e., the final parameter vector will be $\hat{\theta} \odot m$. Since $X\hat{\theta} \odot m = X\mathrm{diag}(\hat{\theta})m$, for finding m, it is sufficient to train over the dataset $X\mathrm{diag}(\hat{\theta})$, with the chosen regularization. In formulas, we minimize

$$\mathcal{L}\big(X\mathrm{diag}(\hat{\theta}), y; m\big) + \lambda\mathrm{R}(m), \quad m \in [0, 1]^d$$

where R is either ℓ_1 or log regularization. We notice that this approach does not retrain the model, but it just performs a selection of the parameters/pixels that can be neglected.

In the experiment, we do not set the ground truth $m^\star$, but we consider acceptable solutions whose accuracy is comparable to the one of standard LG.

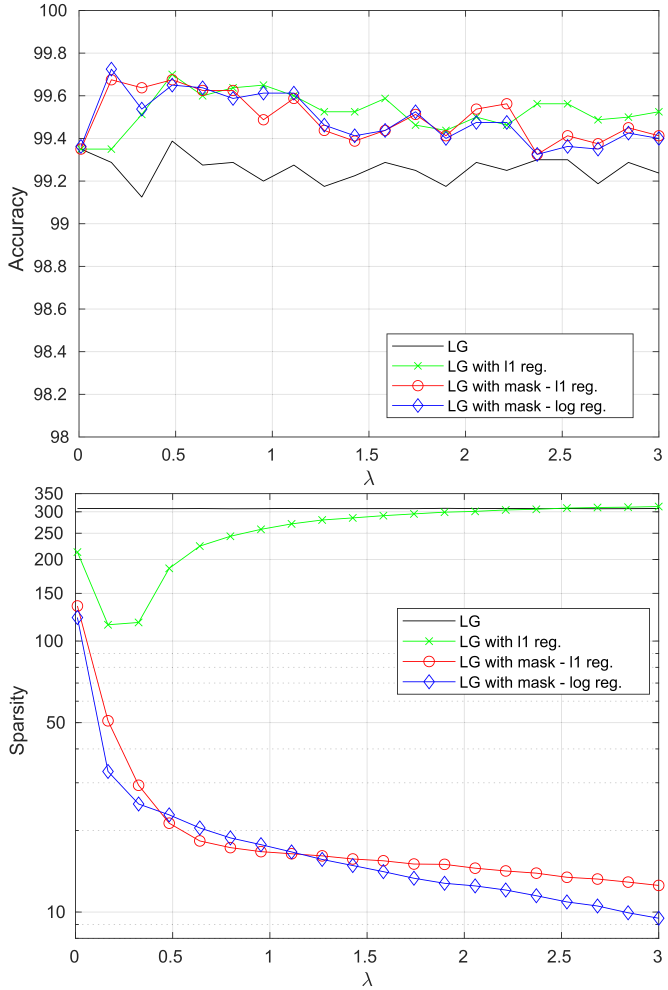

Fig. 3. Binary classification on MNIST dataset with LG accuracy and sparsity with respect to the design parameter λ.

The results of the experiment can be visualized in Fig. 3, where we depict the mean accuracy and sparsity over 100 random runs, with respect to different values of the design parameter λ. In all the approaches, the minimization is performed via gradient descent. More precisely, for the ℓ_1 approach on θ, a subgradient method is used given the nondifferentiability of ℓ_1. As previously discussed, this may not provide sparse solutions, which is clearly visible in the experiment: basically, a larger λ causes large oscillations around zero and the solution is not sparse. This makes this solution not useful, and even less sparse than the original LG. This issue is solved by using the mask: since the relaxed mask is positive, we can use a projected gradient descent, which yields sparse solutions.

In particular, we notice that all the methods achieve comparable accuracy, but the proposed mask ℓ_1/log methods efficiently sparsify the model. As expected from the theoretical results, in most cases, the logarithmic version is more accurate than the ℓ_1 version, by providing sparser models.

In this regard, in Fig. 4(a), we can see an 8-sparse mask obtained via logarithmic regularization. By applying this mask on the eight digits in Fig. 2, we obtain the masked digits in 4(b): even though extremely sparsified, from visual inspection, we can verify that 0 and 1 are distinguishable.

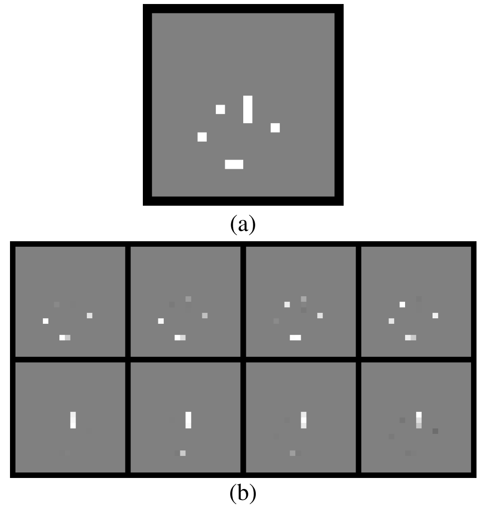

Fig. 4. (a) Eight-sparse mask obtained with logarithmic regularization. (b) Samples of Fig. 2 masked via (a). We can see that eight pixels may be sufficient to distinguish between digits 0 and 1.

V. Experiments

In this section, we perform an experimental evaluation of the proposed method. We first illustrate the experimental settings, then we discuss the experimental performance of the proposed method both in the context of the lottery ticket hypothesis and in network pruning. Then, an ablation study is conducted to evaluate the impact of the various aspects of the proposed technique. Finally, we report the training times of our method.

A. Concave Regularizers

In the experimental evaluation, we consider two concave regularizers, chosen among the most popular ones. The first one is $\mathfrak{r}_1(m_i) = m_i$. As discussed in the previous section, this regularizer corresponds to the ℓ_1-norm restricted to the interval [0, 1] and represents the limit case since it is both convex and concave. The second concave regularizer considered in the experiments is $\mathfrak{r}_\epsilon(m_i) = (\log((m_i + \epsilon)/\epsilon))/(\log((1 + \epsilon)/\epsilon))$, which is strictly convex.

B. Networks and Datasets

We test the chosen concave regularizers on various datasets and architectures. In particular, we focus on image classification. In this context, we consider three architectures: ResNet-20, VGG-11, and WideResNet-20 with 64 convolutional filters in the first block of the network. All these architectures are tested on CIFAR-10, CIFAR-100, and Tiny ImageNet datasets, resulting in nine distinct combinations. In the context of network pruning, we evaluate the performance of the proposed method on CIFAR-10 using the ResNet-56 architecture, which is a standard experiment in this framework.

C. Sparsities

As discussed by [11], sparsities can be divided into three ranges. Trivial sparsities are the lowest sparsities, where even random pruning can achieve full accuracy. Matching sparsities

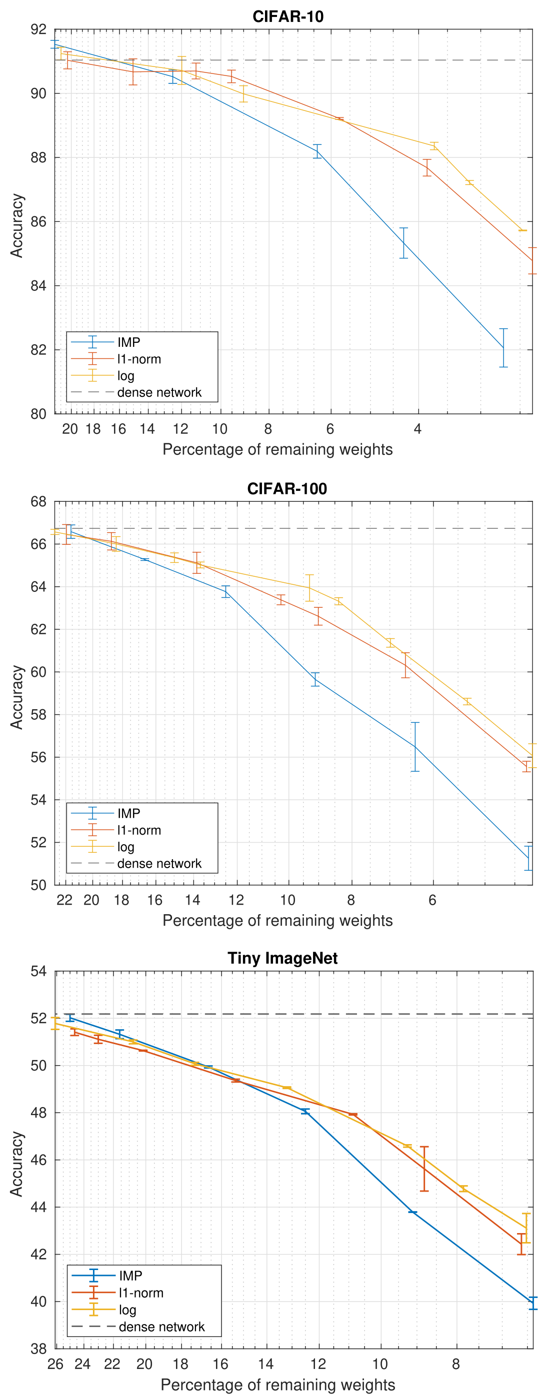

Fig. 5. Test accuracy and sparsity of the subnetworks of ResNet-20 produced by IMP and the proposed method on CIFAR-10, CIFAR-100, and Tiny ImageNet.

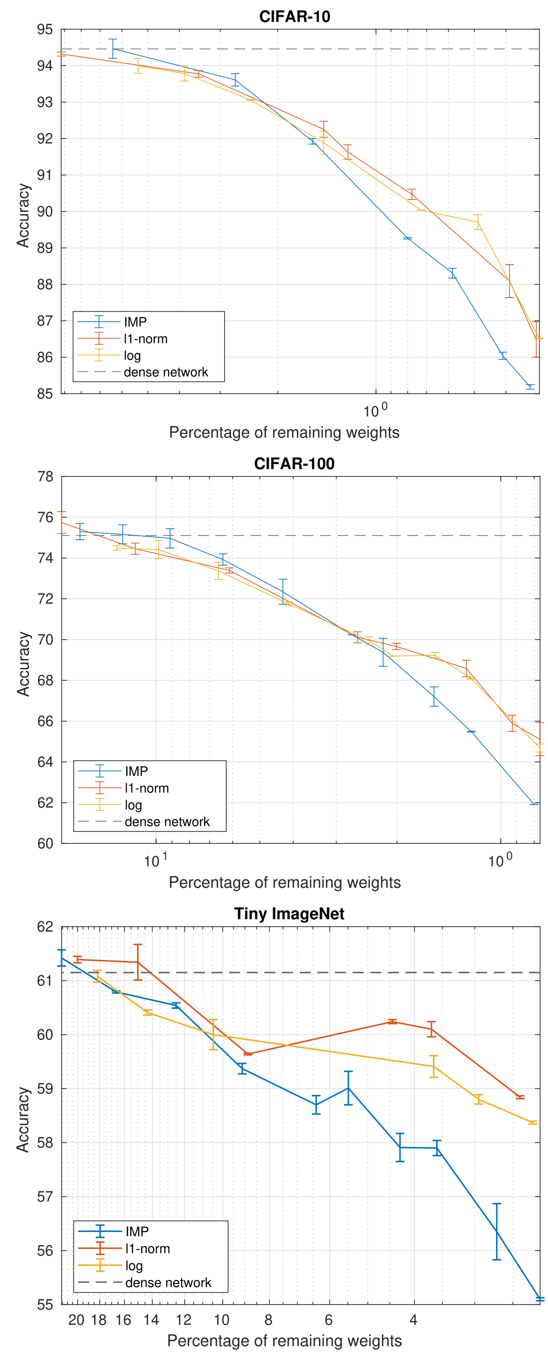

Fig. 6. Test accuracy and sparsity of the subnetworks of WideResNet-20 produced by IMP and the proposed method on CIFAR-10, CIFAR-100, and Tiny ImageNet.

correspond to moderate sparsities, where benchmark methods can approximately reach the accuracy of the full network. Extreme sparsities are the highest sparsities, where the subnetworks cannot reach full accuracy. In these experiments, we consider matching sparsities and the lowest extreme sparsities.

D. Experimental Settings

For all the methods considered in the experimental evaluation, we perform three pruning rounds to identify the sparse subnetwork. In each round, we follow the standard setup proposed in [4], training with SGD, a learning rate of 0.1, a weight decay of 0.0001, and a momentum of 0.9. Each round constitutes 85 epochs for ResNet-20 and 120 for VGG-11 and WideResNet-20, using a batch size of 128. We decay the learning rate by a factor of 10 at epochs 56 and 71 for ResNet-20 and epochs 60 and 90 for VGG-11 and WideResNet-20. The results are averaged on three runs, with different random seeds. For the concave regularizers, we set $\lambda = 3 \times 10^{-6}$ for $\mathrm{r}_1(m_i)$ and $\lambda = 10^{-6}$ for $\mathrm{r}_\epsilon(m_i)$. In addition, for $\mathrm{r}_\epsilon(m_i)$ we set $\epsilon = 0.1$. All the methods

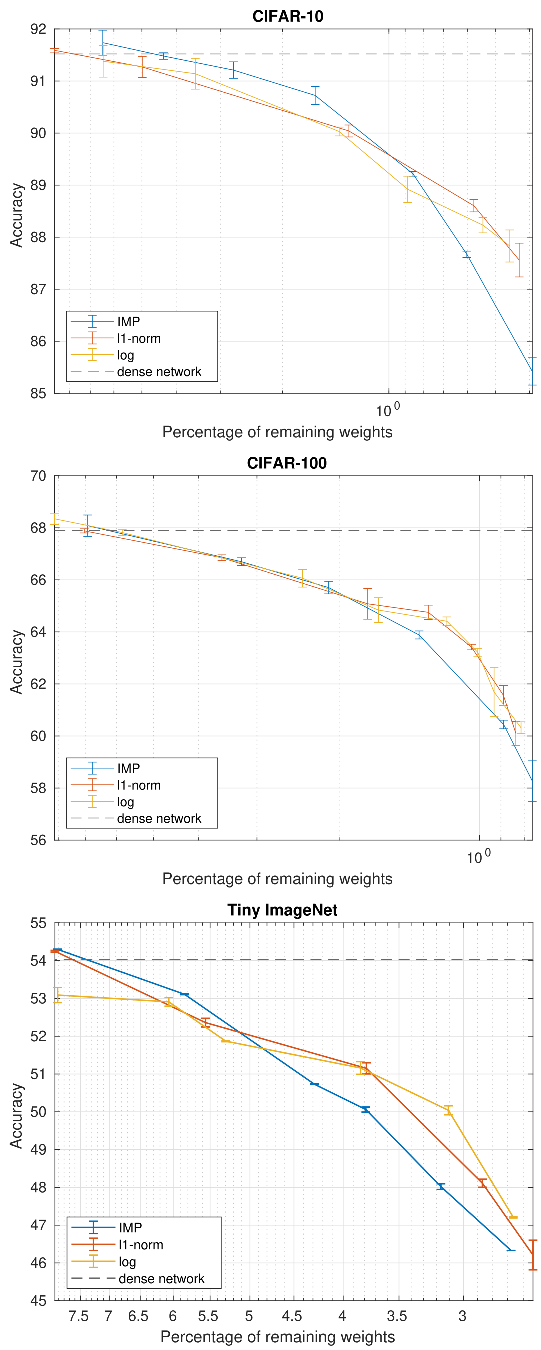

Fig. 7. Test accuracy and sparsity of the subnetworks of VGG-11 produced by IMP and the proposed method on CIFAR-10, CIFAR-100, and Tiny ImageNet.

considered in the experiments were implemented using the publicly available open_lth framework[1] (MIT license). All the experiments were conducted using two NVIDIA TITAN Xp GPUs.

E. Performance Comparison

We consider IMP as a benchmark method. Both the proposed method and IMP are implemented using the same training parameters described in the previous section. Figs. 5–7

[1]https://github.com/facebookresearch/open_lth

TABLE II

COMPARISON AGAINST STATE-OF-THE-ART PRUNING METHODS ON THE CIFAR10 DATASET USING THE RESNET-56 ARCHITECTURE. COLUMNS RESPECTIVELY REPORT THE BASELINE EVALUATION ACCURACY, THE RATE OF UNPRUNED PARAMETERS, THE EVALUATION ACCURACY OF THE PRUNED MODEL, AND THE ACCURACY DIFFERENCE AGAINST THE BASELINE

Method	Baseline (%)	Unpruned (%)	Acc. (%)	Acc. Diff. (%)
GBN-40 [37]	93.10	46.50	93.43	+0.33
GBN-30 [37]	93.10	33.30	93.07	-0.03
Li et. al [38]	93.10	86.30	93.08	-0.02
NISP [39]	93.10	57.40	93.13	+0.03
DCP-A [40]	93.10	29.70	93.09	-0.01
SCOP [41]	93.70	43.70	93.64	-0.06
SFP [42]	93.59	49.40	92.26	-1.33
GAL [43]	93.26	55.20	92.74	-0.52
GAL-0.6 [43]	93.26	88.20	92.98	-0.28
FPGM [44]	93.59	49.4	93.49	-0.10
HRank [18]	93.59	57.60	93.50	-0.09
HRank [18]	93.59	31.90	90.72	-2.87
White-Box [21]	93.26	43.70	93.20	-0.06
AMC [45]	92.80	50.00	91.90	-0.90
SCP [46]	93.69	48.50	93.23	-0.46
LFPC [47]	93.26	47.10	93.24	-0.02
DSA [17]	93.12	46.40	92.92	-0.66
E-Pruner [19]	93.18	71.36	93.10	-0.08
FilterSketch [20]	93.26	79.40	93.65	+0.39
FilterSketch [20]	93.26	58.80	93.19	-0.07
FilterSketch [20]	93.26	28.20	91.20	-2.06
Ours (l1-norm)	93.26	1.79	88.06	-5.02
Ours (l1-norm)	93.26	5.36	92.18	-1.08
Ours (l1-norm)	93.26	9.93	92.99	-0.27
Ours (l1-norm)	93.26	19.34	93.13	-0.13
Ours (l1-norm)	93.26	22.10	93.28	0.02
Ours (Log)	93.26	1.66	87.98	-5.28
Ours (Log)	93.26	5.12	92.17	-1.09
Ours (Log)	93.26	9.91	93.01	-0.25
Ours (Log)	93.26	18.47	93.11	-0.15
Ours (Log)	93.26	23.2	93.30	0.04

show the results for ResNet-20, WideResNet-20, and VGG-11, respectively. We can observe that the proposed method with the concave regularizers reaches performance comparable to the ones of IMP at matching sparsities, but significantly outperforms IMP at higher sparsity, especially on the ResNet-20 architecture. The two concave regularizers exhibit similar performance in most of the experiments. However, in some cases, such as the experiments on ResNet-20 depicted in Fig. 5, the strictly convex regularizer $r_\epsilon(m_i)$ can significantly improve the performance with respect to $r_1(m_i)$.

F. Comparison Against State-of-the-Art Pruning Methods

Even though the main focus of this work is the lottery ticket hypothesis where we aim to find sparse trainable networks, we also compare our work against state-of-the-art pruning methods that consider a dense-to-sparse approach. In this case, we evaluate the performance of the proposed method using a standard experiment for pruning methods, i.e., image classification on the CIFAR10 dataset using the ResNet-56 architecture. Results in Table II show that our method greatly outperforms competing methods: we are able to improve over the baseline accuracy with an unpruned ratio as low as 23%, which indicates that more than 77% of the parameters have been pruned.

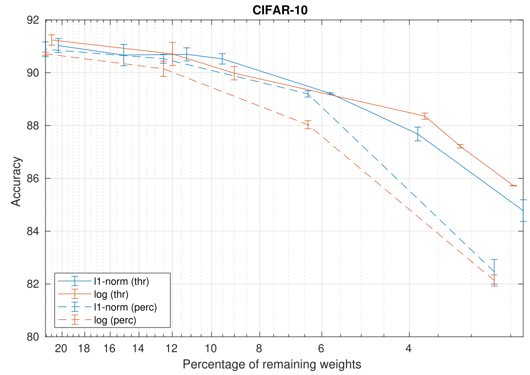

Fig. 8. Test accuracy and sparsity of the subnetworks of ResNet-20 produced by the proposed method on CIFAR-10. Solid lines correspond to the case where we prune the weights whose mask values are below a given threshold (soft approach). Dashed lines correspond to the case where at each round we prune a fixed percentage of weights based on the magnitude of the mask value (hard approach).

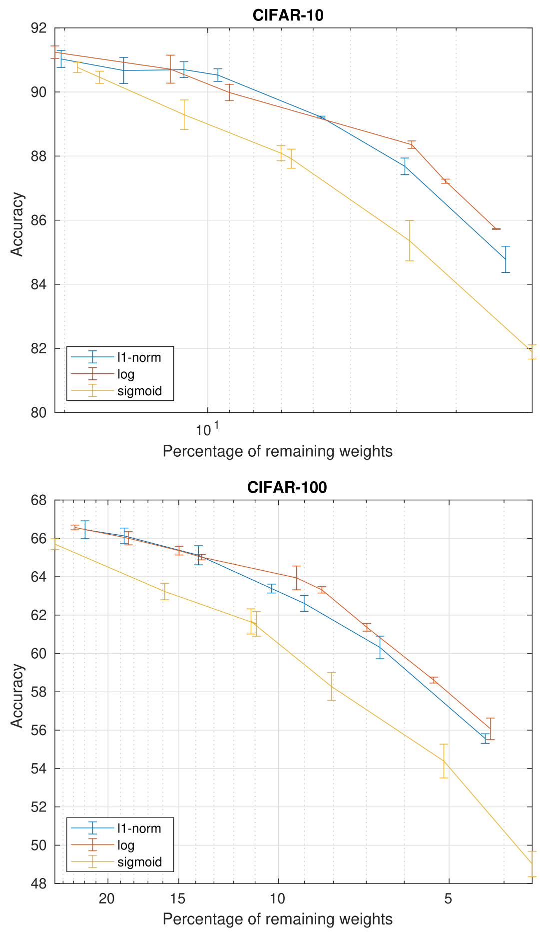

Fig. 9. Test accuracy and sparsity of the subnetworks of ResNet-20 produced the proposed method using concave regularizer or sigmoid reparameterization on CIFAR-10 (left) and CIFAR-100 (right).

G. Ablation Study

In this section, we evaluate the impact of some aspects of the proposed method. First, we investigate the impact of setting a threshold instead of removing a fixed percentage of parameters. In the first case, we remove the parameters whose mask values are below a given threshold, at the end of each round. In contrast, in the second case, a constant percentage of weights is removed at each round based on the magnitude of the mask values. Fig. 8 shows the results of this comparison for ResNet-20 on CIFAR-10. We can observe that setting a threshold can significantly improve the performance of the proposed method.

We then evaluate the importance of using concave regularizers, comparing them against a different regularization technique. We consider a sigmoid reparameterization $m = \sigma(\beta s) = (1/(1 + e^{-\beta s}))$, as proposed in [12]. The parameter β is defined as in [12], where is updated at each epoch using an exponential schedule $\beta^{(t)} = (\beta^{(T)})^{t/T}$ for epoch $t = 0, \dots, T$, where we set $\beta = 200$. The results for ResNet-20 on CIFAR-10 and CIFAR-100 are shown in Fig. 9. We can observe that the proposed concave regularizers can provide significantly better performance than the sigmoid reparameterization.

TABLE III
TRAINING TIMES ON CIFAR-10 OF THE ARCHITECTURES CONSIDERED IN THE EXPERIMENTS

Architecture	Time to find the sparse network	Training time of the sparse network
ResNet-20	91 minutes	27 minutes
WideResNet-20	168 minutes	54 minutes
VGG-11	119 minutes	40 minutes
ResNet-56	244 minutes	67 minutes

H. Training Times

In Table III, we indicate the training times on CIFAR-10 of the architectures considered in the experiments. Table III shows both the time for finding the sparse architecture and the training time of the sparse network.

VI. CONCLUSION

In this article, we proposed a novel iterative method for identifying matching tickets, i.e., for extracting subnetworks that, when trained in isolation, achieve accuracy comparable to the dense networks that contain them. The method is based on the introduction of a binary mask, which is relaxed to a continuous variable and penalized by concave regularization to promote its sparsity. The use of strictly concave regularization is quite novel in deep learning. We provide theoretical results that substantiate its effectiveness with respect to other methods. Experiments on different problems and architectures show that the proposed approach is valuable with respect to the IMP method. Future work will be devoted to optimizing the tuning of the hyperparameters and to the reduction of the numerical complexity, by considering early-bird approaches and untrained models as in the strong lottery ticket framework.

REFERENCES

[1] Y. LeCun, J. Denker, and S. Solla, “Optimal brain damage,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 2, 1990.
[2] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both weights and connections for efficient neural network,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2015, pp. 1135–1143.

[3] C. Louizos, M. Welling, and D. P. Kingma, "Learning sparse neural networks through ℓ_0 regularization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[4] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019.

[5] T.-J. Yang, Y.-H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5687–5695.

[6] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.

[7] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" in *Proc. Mach. Learn. Syst. (MLSys*, vol. 2, 2020, pp. 129–146.

[8] M. H. Zhu and S. Gupta, "To prune, or not to prune: Exploring the efficacy of pruning for model compression," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[9] Y. Yang, Y. Yuan, A. Chatzimichailidis, R. J. van Sloun, L. Lei, and S. Chatzinotas, "ProxSGD: Training structured neural networks under regularization and constraints," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020.

[10] J. Yun, A. C. Lozano, and E. Yang, "Adaptive proximal gradient methods for structured neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 34, 2021, pp. 24365–24378.

[11] J. Frankle, G. K. Dziugaite, D. Roy, and M. Carbin, "Linear mode connectivity and the lottery ticket hypothesis," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 3259–3269.

[12] P. Savarese, H. Silva, and M. Maire, "Winning the lottery with continuous sparsification," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 11380–11390.

[13] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2755–2763.

[14] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019.

[15] E. Tartaglione, A. Bragagnolo, F. Odierna, A. Fiandrotti, and M. Grangetto, "SeReNe: Sensitivity-based regularization of neurons for structured sparsity in neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 7237–7250, Dec. 2022.

[16] H. Salehinejad and S. Valaee, "EDropout: Energy-based dropout and pruning of deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 10, pp. 5279–5292, Oct. 2022.

[17] X. Ning, T. Zhao, W. Li, P. Lei, Y. Wang, and H. Yang, "DSA: More efficient budgeted pruning via differentiable sparsity allocation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 592–607.

[18] M. Lin et al., "HRank: Filter pruning using high-rank feature map," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1529–1538.

[19] M. Lin et al., "Network pruning using adaptive exemplar filters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 7357–7366, Dec. 2022.

[20] M. Lin et al., "Filter sketch for network pruning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 7091–7100, Dec. 2022.

[21] Y. Zhang et al., "Carrying out CNN channel pruning in a white box," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 10, pp. 7946–7955, Oct. 2023.

[22] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Stat. Soc., B, Methodol.*, vol. 58, no. 1, pp. 267–288, Jan. 1996.

[23] E. J. Candès, M. B. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, nos. 5–6, pp. 877–905, 2008.

[24] J. Woodworth and R. Chartrand, "Compressed sensing recovery via nonconvex shrinkage penalties," *Inverse Problems*, vol. 32, no. 7, Jul. 2016, Art. no. 075004.

[25] C. H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *Ann. Statist.*, vol. 38, no. 2, pp. 894–942, Apr. 2010.

[26] S. Foucart and M.-J. Laui, "Sparsest solutions of underdetermined linear systems via ℓ_q-minimization for $0< q\leq 1$," *Appl. Comput. Harmon. Anal.*, vol. 26, pp. 395–407, 2009.

[27] I. Selesnick, "Sparse regularization via convex analysis," *IEEE Trans. Signal Process.*, vol. 65, no. 17, pp. 4481–4494, Sep. 2017.

[28] V. Cerone, S. M. Fosson, D. Regruto, and A. Salam, "Sparse learning with concave regularization: Relaxation of the irrepresentable condition," in *Proc. 59th IEEE Conf. Decis. Control (CDC)*, Dec. 2020, pp. 396–401.

[29] F. Wen, L. Chu, P. Liu, and R. C. Qiu, "A survey on nonconvex regularization-based sparse and low-rank recovery in signal processing, statistics, and machine learning," *IEEE Access*, vol. 6, pp. 69883–69906, 2018.

[30] I. Bayram, "On the convergence of the iterative shrinkage/thresholding algorithm with a weakly convex penalty," *IEEE Trans. Signal Process.*, vol. 64, no. 6, pp. 1597–1608, Mar. 2016.

[31] P. Gong, C. Zhang, Z. Lu, J. Z. Huang, and J. Ye, "A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2013, pp. 37–45.

[32] K. Bredies, D. A. Lorenz, and S. Reiterer, "Minimization of non-smooth, non-convex functionals by iterative thresholding," *J. Optim. Theory Appl.*, vol. 165, no. 1, pp. 78–112, Apr. 2015.

[33] M. Hong, Z. Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM J. Optim.*, vol. 26, no. 1, pp. 337–364, Jan. 2016.

[34] V. Ramanujan, M. Wortsman, A. Kembhavi, A. Farhadi, and M. Rastegari, "What's hidden in a randomly weighted neural network?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11890–11899.

[35] E. Malach, G. Yehudai, S. Shalev-Schwartz, and O. Shamir, "Proving the lottery ticket hypothesis: Pruning is all you need," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 119, 2020, pp. 6682–6691.

[36] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

[37] Z. You, K. Yan, J. Ye, M. Ma, and P. Wang, "Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks," in *Proc. Adv. Neural Inform. Process. Syst. (NeurIPS)*, vol. 32, 2019.

[38] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. Peter Graf, "Pruning filters for efficient ConvNets," 2016, *arXiv:1608.08710*.

[39] R. Yu et al., "NISP: Pruning networks using neuron importance score propagation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9194–9203.

[40] Z. Zhuang et al., "Discrimination-aware channel pruning for deep neural networks," in *Proc. Adv. Neural Inform. Process. Syst. (NeurIPS)*, vol. 31, 2018.

[41] Y. Tang et al., "SCOP: Scientific control for reliable neural network pruning," in *Proc. Adv. Neural Inform. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 10936–10947.

[42] Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang, "Soft filter pruning for accelerating deep convolutional neural networks," 2018, *arXiv:1808.06866*.

[43] S. Lin et al., "Towards optimal structured CNN pruning via generative adversarial learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2790–2799.

[44] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter pruning via geometric median for deep convolutional neural networks acceleration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4340–4349.

[45] Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han, "AMC: Automl for model compression and acceleration on mobile devices," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 784–800.

[46] M. Kang and B. Han, "Operation-aware soft channel pruning using differentiable masks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5122–5131.

[47] Y. He, Y. Ding, P. Liu, L. Zhu, H. Zhang, and Y. Yang, "Learning filter pruning criteria for deep convolutional neural networks acceleration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2009–2018.

Giulia Fracastoro (Member, IEEE) received the Ph.D. degree in electronic and telecommunications engineering from Politecnico di Torino, Turin, Italy, in 2017.

She is currently an Assistant Professor with the Department of Electronics and Telecommunications (DET), Politecnico di Torino. Her research interests include graph signal processing, system identification, and deep learning.

Dr. Fracastoro is an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING and IEEE TRANSACTIONS ON SIGNAL AND INFORMATION PROCESSING OVER NETWORKS. She was a recipient of IEEE International Conference on Image Processing (ICIP) 2019 Best Paper Award, IEEE Picture Coding Symposium (PCS) 2016 Best Paper Award, and IEEE International Conference on Image Processing (ICIP) 2015 Best Student Paper Award.

Sophie M. Fosson (Member, IEEE) received the M.Sc. degree in applied mathematics from the Politecnico di Torino, Turin, Italy, in 2005, and the Ph.D. degree in mathematics for the industrial technologies from Scuola Normale Superiore di Pisa, Pisa, Italy, in 2011. From 2012 to 2016, she was a Post-Doctoral Associate with the Department of Electronics and Telecommunications, Politecnico di Torino, where she is currently an Assistant Professor with the Department of Control and Computer Engineering. She visited the Centre Tecnològic de Telecomunicacions de Catalunya, Barcelona, Spain, in 2013, 2014, and 2016. She was a Researcher with Istituto Superiore Mario Boella, Turin, in 2017. She is Associate Editor of IEEE CONTROL SYSTEMS LETTERS. Her main research interests include sparse optimization, machine learning, system identification, control, and cyber-physical systems.

Andrea Migliorati received the M.Sc. degree in communication technologies and multimedia in 2015 from Università degli Studi di Brescia, Brescia, Italy and the Ph.D. degree in telecommunication engineering in 2019 after a three-year collaboration between Università di Brescia and Telecom Italia's Joint Open Lab in Turin, Italy.

He is an Assistant Professor at the Department of Electronics and Telecommunications, Politecnico di Torino, Turin, he works with deep learning techniques for image analysis, adversarial robustness, biometric verification, encryption algorithms for EO on-board imaging, and new techniques for training binarized deep neural networks.

Giuseppe C. Calafiore (Fellow, IEEE) was a Senior Fellow with the Institute of Pure and Applied Mathematics (IPAM), University of California at Los Angeles, Los Angeles, CA, USA, in 2010. He is currently a Full Professor with DET, Politecnico di Torino, Turin, Italy, where he coordinates the Control Systems and Data Science group. He was a Professor of Mechanical Engineering at CECS, VinUniversity, Hanoi, Vietnam, in 2023. He was an Associate Fellow of the IEIIT-CNR, Turin. He held visiting positions at the Information Systems Laboratory (ISL), Stanford University, Stanford, CA, USA, in 1995; the Ecole Nationale Supérieure de Techniques Avancées (ENSTA), Paris, France, in 1998; and the University of California at Berkeley, Berkeley, CA, USA, in 1999, 2003, 2007, 2017–2019, and 2021, where he co-taught a master course on Financial Data Science. He is the author of about 210 journal and conference proceedings papers, and of eight books. His research interests are in the fields of convex optimization, identification and control of uncertain systems, with applications ranging from finance and economic systems to robust control, machine learning, and data science.

Dr. Calafiore received IEEE Control System Society "George S. Axelby" Outstanding Paper Award in 2008.