From Sparse Bayesian Learning to Deep Recurrent Nets

Hao He
Peking University

Bo Xin
Microsoft Research, Beijing

David Wipf
Microsoft Research, Beijing

I. Introduction

We begin with the canonical sparse estimation problem

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_0, \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is an observed vector, $\boldsymbol{\Phi} \in \mathbb{R}^{n\times m}$ is some known dictionary of basis vectors with $m > n$, $\|\cdot\|_0$ denotes the ℓ_0 sparsity-promoting norm, and λ is a trade-off parameter. Although crucial to many applications [1], [2], [3], [4], [5], [6], solving (1) is NP-hard. Popular approximations include convex relaxations such as ℓ_1-norm regularization [7], [8], [9] and iterative hard-thresholding (IHT) [10], [11]. However, a core weakness underlies them all: *If the columns of $\boldsymbol{\Phi}$ are highly correlated, then estimation of $\boldsymbol{x}^*$ may be poor.*

Interestingly, many existing sparse estimation implementations involve an update rule comprised of a fixed, linear filter followed by a non-linear activation function that promotes sparsity, both features of typical deep neural network layers. Consequently, algorithm execution can be interpreted as passing an input through a deep network with constant filters (dependent on $\boldsymbol{\Phi}$) at every layer [12], [13], which is also tantamount to a simple form of recurrent network. This association immediately suggests that we consider substituting discriminatively learned filters in place of those inspired by the original sparse recovery algorithm. For example, it has been argued that, given access to a sufficient number of $\{\boldsymbol{x}^*, \boldsymbol{y}\}$ pairs, a trained network may be capable of producing quality sparse estimates with a few number of layers [12], [14], [15], [16].

In each of these cases however, the initial archetype is a sparse estimation algorithm known to be highly sensitive to data correlations, e.g., convex relaxations, IHT, etc. But if our ultimate goal is to learn a new 'algorithm' that efficiently compensates for structure in $\boldsymbol{\Phi}$, it seems reasonable to invoke iterative approaches known *a priori* to handle such correlations directly as our template for learned network layers. One important example is sparse Bayesian learning (SBL) [17], which has been shown to solve (1) even in cases where $\boldsymbol{\Phi}$ displays strong correlations [18]. Herein we demonstrate that, when judiciously unfolded, SBL iterations can be formed into variants of long short-term memory (LSTM) cells, one of the more popular recurrent deep neural network architectures [19], [20]. The resulting network dramatically outperforms existing methods in solving (1) with a minimal computational budget.

II. Connecting SBL to LSTM Networks and Beyond

Although not originally derived this way, SBL can be implemented using a form of iterative reweighted ℓ_1-norm optimization that exposes interesting connections with deep learning and recurrent LSTM cells. In general, if we replace the ℓ_0 norm from (1) with any smooth approximation $g(|\boldsymbol{x}|)$, where g is a concave, non-decreasing function and $|\cdot|$ applies elementwise, then global convergence to some stationary point can be guaranteed using iterations of the form

$$\boldsymbol{x}^{(k+1)} \;\rightarrow\; \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\|_2^2 + \lambda \sum_i w_i^{(k)}|x_i| \tag{2}$$

$$w_i^{(k+1)} \;\rightarrow\; \partial g(\boldsymbol{u})/\partial u_i|_{u_i=\left|x_i^{(k+1)}\right|}, \quad \forall i. \tag{3}$$

In the context of SBL, there is no closed-form $w_i^{(k+1)}$ update except in special cases [21]. But if we allow for additional latent structure akin to the memory unit of LSTM cells, a viable recurrency emerges for computing these weights and elucidating their effectiveness in dealing with correlated dictionaries. In particular we have:

Proposition 1. *The weights $\boldsymbol{w}^{(k+1)}$ invoked by the iterative reweighted ℓ_1 form of SBL satisfy*

$$\left(w_i^{(k+1)}\right)^2 = \min_{\boldsymbol{z}:\mathit{supp}[\boldsymbol{z}]=\mathit{supp}[\boldsymbol{\gamma}]} \frac{1}{\lambda}\|\boldsymbol{\phi}_i - \boldsymbol{\Phi}\boldsymbol{z}\|_2^2 + \sum_{j\in\mathit{supp}[\boldsymbol{\gamma}]} \frac{z_j^2}{\gamma_j} \tag{4}$$

for all i, where $\gamma_j \triangleq \left[w_j^{(k)}\right]^{-1}\left|x_j^{(k+1)}\right|$.

Unlike traditional iterative reweighted sparsity algorithms [22], with SBL we see that the i-th weight w_i is not dependent solely on the value of the i-th coefficient x_i, but rather on *all* the latent variables $\boldsymbol{\gamma}$ and ultimately prior-iteration weights $\boldsymbol{w}$ as well, linking the fate of each sparse coefficient such that correlation structure can be properly accounted for in a progressive fashion. More concretely, from (4) it is immediately apparent that if $\boldsymbol{\phi}_i \approx \boldsymbol{\phi}_{i'}$ for some indeces i and i' (meaning a large degree of correlation), then it is highly likely that $w_i \approx w_{i'}$. This is simply because the regularized residual error that emerges from solving (4) will tend to be quite similar when $\boldsymbol{\phi}_i \approx \boldsymbol{\phi}_{i'}$. In this situation, a suboptimal solution will not be prematurely enforced by weights with large, spurious variance across a correlated group of basis vectors. A crucial exception occurs when $\boldsymbol{\gamma}$ is highly sparse, or nearly so, in which case there are limited degrees of freedom with which to model small differences in each $\boldsymbol{\phi}_i$; however, such cases can generally only occur when we are in the neighborhood of ideal, maximally sparse solutions, when different weights are actually desirable for resolving the final sparse estimates.

Importantly, the additional latent variables, when structured as we have done in (4), also closely mimic the behavior of the latent state in an LSTM cell, and the associated gating mechanisms that allow for incrementally storing or updating learned representations (akin to incrementally accruing the correct sparsity profile). In fact, both the required optimizations from (2) and (4) can be implemented/approximated with simple recurrent networks based on iterative thresholding/shrinkage algorithms, while their combination requires an LSTM-like architecture or gated feedback extensions [20]. These can be viewed as learning momentum-like terms [7] and/or optimal transitions between inner and outer optimization loops. For detailed correspondences and further technical analyses see [23].

III. Broader Implications

Overall, the progression from iterative thresholding algorithms to SBL mirrors the progression from simple, vanilla-flavored recurrent networks to organizations of complex LSTM cells. But from a much wider perspective, this correspondence suggests that the micro-management of multi-loop iterative algorithm trajectories can actually be learned, as opposed to existing handcrafted algorithms with fixed inner/outer loop scheduling that may be suboptimal in terms of both speed and estimation accuracy. For more discussion, please see [23].

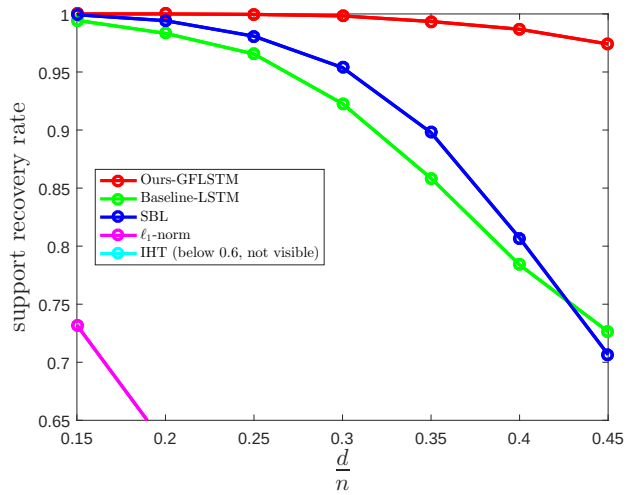

Fig. 1. *Empirical results demonstrating the utility of our adaptations.* Comparisons follow the set-up from [16][Section 7.2], where the goal is to recover maximally sparse (or minimal ℓ_0 norm) feasible solutions to some system $\boldsymbol{y} = \boldsymbol{\Phi x}$, with columns of $\boldsymbol{\Phi}$ highly correlated by design. This is equivalent to solving (1) with λ small. Training pairs $\{\boldsymbol{x}^*, \boldsymbol{y}\}$ are randomly generated using the linear forward model, and a network is trained to learn the inverse mapping from $\boldsymbol{y} = \boldsymbol{\Phi x}^*$ to $\boldsymbol{x}^*$. We invoke a particular gated feedback (GF) LSTM architecture from [20] adapted to reflect unfolded SBL iterations, followed by a final softmax layer for predicting support patterns. Two stacked recurrent layers are utilized, to model inner and outer algorithmic loops, which are unfolded for only 11 iterations leading to efficient runtime with test data (see below). Recovery results are shown as $d \triangleq \|\boldsymbol{x}^*\|_0$ is varied. The accuracy metric, as defined in [16], measures the percentage of correctly identified supporting elements of $\boldsymbol{x}^*$ contained among the largest d values of the network output. We observe that our approach significantly outperforms both a rudimentary LSTM network we explore in [16], which represents the best existing learned deep network structure for this task (albeit without rigorous motivation), as well as SBL, the best pure optimization-based approach we have found for handling this type of correlated dictionary. Other approaches, such as ℓ_1 minimization, IHT, or learned variants of these [12], [15], have a success rate below 0.7 and hence do not appear in the figure. Additional testing conditions reveal similar improvements afforded by the proposed GFLSTM approach [23]. Note also that a recent interesting modification of approximate message passing can handle certain specialized forms of dictionary correlation [24]; however, the approach does not work with the types of strong correlation we have utilized for our experiments, with results inferior to the ℓ_1 norm solution.

TABLE I

AVERAGE PER-SAMPLE RUNTIMES (IN SECONDS) TO PRODUCE SPARSE ESTIMATES. WITH THE PROPOSED NETWORK, ONCE TRAINING IS COMPLETED, THE FINAL NETWORK IS EXTREMELY EFFICIENT AT PRODUCING SPARSE ESTIMATES GIVEN NOVEL INPUTS.

	IHT	ℓ_1 norm	SBL	Ours-GFLSTM
runtime(sec)	0.0329	0.0766	0.1144	2.481×10^{-5}

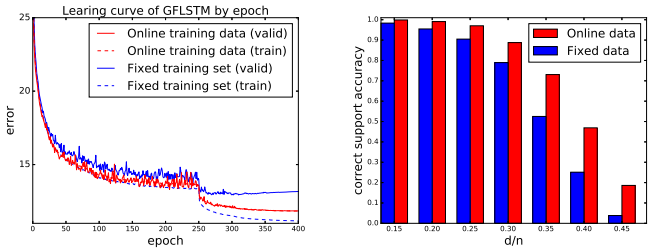

Fig. 2. *A useful training heuristic. Left*: When training with a fixed-sized dataset, as existing learning approaches to sparse estimation do [12], [15], [16], there is always the risk of overfitting (note the gap between training and validation sets with the blue learning curves). However, since we are free to generate online as much training data as we want, at every epoch we can always use a new, unseen batch. This simple strategy completely closes the gap (red curves) with negligible computational overhead. *Right*: Resulting improvement in performance, as measured by the percentage of trials whereby the entire support pattern is correctly estimated, a complementary evaluation metric to the one described above. Please see [23] for more details.

REFERENCES

[1] S. Baillet, J. Mosher, and R. Leahy, "Electromagnetic brain mapping," *IEEE Signal Processing Magazine*, pp. 14–30, Nov. 2001.

[2] E. Candès and T. Tao, "Decoding by linear programming," *IEEE Trans. Information Theory*, vol. 51, no. 12, 2005.

[3] S. Cotter and B. Rao, "Sparse channel estimation via matching pursuit with application to equalization," *IEEE Trans. on Communications*, vol. 50, no. 3, 2002.

[4] M. Figueiredo, "Adaptive sparseness using Jeffreys prior," *NIPS*, 2002.

[5] S. Ikehata, D. Wipf, Y. Matsushita, and K. Aizawa, "Robust photometric stereo using sparse regression," in *CVPR*, 2012.

[6] D. Malioutov, M. Çetin, and A. Willsky, "Sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE Trans. Signal Processing*, vol. 53, no. 8, 2005.

[7] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sciences*, vol. 2, no. 1, 2009.

[8] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[9] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. of the Royal Statistical Society*, 1996.

[10] T. Blumensath and M. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, 2009.

[11] ——, "Normalized iterative hard thresholding: Guaranteed stability and performance," *IEEE J. Selected Topics Signal Processing*, vol. 4, no. 2, 2010.

[12] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *ICML*, 2010.

[13] J. Hershey, J. L. Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," *arXiv preprint arXiv:1409.2574v4*, 2014.

[14] P. Sprechmann, A. Bronstein, and G. Sapiro, "Learning efficient sparse and low rank models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, 2015.

[15] Z. Wang, Q. Ling, and T. Huang, "Learning deep ℓ_0 encoders," *arXiv preprint arXiv:1509.00153v2*, 2015.

[16] B. Xin, Y. Wang, W. Gao, and D. Wipf, "Maximal sparsity with deep networks?" *arXiv preprint arXiv:1605.01636*, 2016.

[17] M. Tipping, "Sparse Bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, 2001.

[18] D. Wipf, "Sparse estimation with structured dictionaries," *Advances in Nerual Information Processing 24*, 2012.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, 1997.

[20] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," in *International Conference on Machine Learning*, 2015.

[21] D. Wipf and S. Nagarajan, "Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions," *Journal of Selected Topics in Signal Processing (Special Issue on Compressive Sensing)*, vol. 4, no. 2, April 2010.

[22] E. Candès, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, no. 5, pp. 877–905, 2008.

[23] H. He, B. Xin, and D. Wipf, "Towards learning a richer class of multi-loop iterative algorithms," *Microsoft Research Technical Report*, 2017.

[24] S. Rangan, P. Schniter, and A. Fletcher, "Vector approximate message passing," *arXiv preprint arXiv:1610.03082*, 2016.