DEMYSTIFYING DEEP LEARNING: A GEOMETRIC APPROACH TO ITERATIVE PROJECTIONS

Ashkan Panahi[†], *Hamid Krim*[†], *Liyi Dai*[†#*]

[†] Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, 27606
The U.S. Army Research Office, Durham, NC 27709

ABSTRACT

Parametric approaches to Learning, such as deep learning (DL), are highly popular in nonlinear regression, in spite of their extremely difficult training with their increasing complexity (e.g. number of layers in DL). In this paper, we present an alternative semi-parametric framework which foregoes the ordinarily required feedback, by introducing the novel idea of geometric regularization. We show that certain deep learning techniques such as residual network (ResNet) architecture are closely related to our approach. Hence, our technique can be used to analyze these types of deep learning. Moreover, we present preliminary results which confirm that our approach can be easily trained to obtain complex structures.

Index Terms— supervised learning, back propagation, geometric approaches

1. INTRODUCTION

Learning a nonlinear function through a finite number of input-output observations is a fundamental problem of supervised machine learning, and has wide applications in science and engineering. From a statistical vantage point, this problem entails a regression procedure which, depending on the nature of the underlying function, may be linear or nonlinear. In the past few decades, there has been a flurry of advances in the area of nonlinear regression [1]. Deep learning is perhaps one of the most well-known approaches with a promising and remarkable performance in great many applications.

Deep learning has a number of distinctive advantages: 1. It relies on a parametric description of functions that are easily computable. Once the parameters (weights) of a deep network are set, the output can be rapidly computed in a feed forward fashion by a few iterations of affine and elementwise nonlinear operations; 2. it can avoid over-parametrization by adjusting the architecture (number of parameters) of the network, hence providing control over the generalization power of deep learning. Finally, deep networks have been observed to be highly flexible in expressing complex and highly nonlinear functions [2, 3]. There are, however, a number of challenges associated with deep learning, chief among them is that of obtaining the exact assessment of their expressive power which remains to this day, an open problem. An important exception is the single-layer network for which the so called universal approximation property (UAP) has been established for some time[4, 5], and which is clearly a highly desirable property. Another practical difficulty with deep learning is that the output becomes unproportionally sensitive to the parameters of different layers, making it, from an optimization perspective, extremely difficult to train [6]. A recent solution is the so-called residual network (ResNet) learning, which introduces bridging branches to the conventional deep learning architecture [7]. In this paper, we address the above issues by proposing a different perspective on learning with a substantially different architecture, which totally forgoes any feedback. Specifically, we propose an interative foward projection in lieu of back propagation to update parameters. As such, this may rapidly yield an over-parametrized system, we restrict each layer to perform an "incremental" update on the data, as approximately captured by the realization of a differential equation, we refer to as geometric regularization, as discussed in Section 2.1. The formulation of this geometric regularization allows us to tie the analysis of deep networks to differential geometry. The study in [8] notices this relation, but adopts a different approach. In particular, we conjecture a converse of the celebrated Frobenius integrability theorem, which potentially proves a universal approximation property for a family of modified deep ResNets. We also present preliminary results in Section 5, and show that foregoing back propagation in a neural network does not greatly limit the expressive power of deep networks, and in fact potentially decreases their training effort, dramatically.

*This work is partially supported by the U.S. Army Research Office under agreement W911NF-16-2-0005

2. MMSE ESTIMATION BY GEOMETRIC REGULARIZATION

For the sake of generality, we consider a C^1 Banach manifold[1] $\mathcal{F}$ of functions $f : \mathbb{R}^n \to \mathbb{R}^m$, where n, m are the dimensions of the data and label vectors, respectively, and each element $f \in \mathcal{F}$ represents a candidate model between the data and the labels. The arbitrary choice of $\mathcal{F}$ allows one to impose structural properties on the models. Due to space limitation and for clarity sake, we just focus on the simpler case of $\mathcal{F} = \mathcal{L}^2$, i.e. the space of square integrable functions, and defer further generalizations to a later publication. Moreover, consider a probability space (Ω, Σ, μ), and two random vectors $\mathbf{x} : \Omega \to \mathbb{R}^n$ and $\mathbf{y} : \Omega \to \mathbb{R}^m$ representing statistical information about the data. As samples $(\mathbf{x}_t, \mathbf{y}_t)$ for $t = 1, 2, \ldots, T$ of $\mathbf{x}, \mathbf{y}$ are often provided, in which case their empirical distribution is used.

We consider the supervised learning problem by minimizing the following mean square error (MSE),

$$L(f) = \mathbb{E}\left[\|f(\mathbf{x}) - \mathbf{y}\|_2^2\right], \tag{1}$$

where $\mathbb{E}[.]$ denotes expectation. For observed samples $(\mathbf{x}_t, \mathbf{y}_t)$, this criterion simplifies to

$$\min_{f \in \mathcal{F}} \frac{1}{T} \sum_{t=1}^{T} \|f(\mathbf{x}_t) - \mathbf{y}_t\|_2^2. \tag{2}$$

In practice, the statistical assumptions in Eq. (2) are highly underdetermined and minimization of MSE (MMSE) leads to undesired solutions. To cope with this, additional constraints are considered to tame the problem by way of regularization. For example the set $\mathcal{F}$ can be restricted to a (finite-dimensional) smooth sub-manifold. This is an implicit fact in parametric approaches, such as deep neural networks.

2.1. Geometric Regularization

We introduce a more general type of regularization, which also includes parametric restriction. Our generalization is inspired by the observation that standard (smooth) optimization techniques such as gradient descent, are based on a realization of a differential equation of the following form,

$$\frac{\mathrm{d}f_\tau}{\mathrm{d}\tau} = \phi(f_\tau), \tag{3}$$

where $\phi(f) \in T_f$ is a tangent vector of $\mathcal{F}$ at f. The resulting solution is typically in an iterative form, as follows,

$$f_{t+1} = f_t + \mu_t \phi(f_t), \tag{4}$$

where μ_t is the step size at iteration $t = 0, 1, 2, \ldots$. The tangent vector $\phi(f_\tau)$ is often selected as a descent direction, where according to Eq. (1), $\mathrm{d}L/\mathrm{d}\tau < 0$.

[1] A Banach manifold is an infinite-dimensional generalization of a conventional differentiable manifold [9].

For geometric regularization, we restrict the choice of the tangent vector to a closed cone $C_f \subseteq T_f$ in the tangent space. In the case of function estimation, where $\mathcal{F}$ and hence the tangent space T_f, is infinite dimensional, we adopt a parametric definition of C_f by restricting the tangent vector to a finite dimensional space. However, this might not restrict the function to a finite dimensional submanifold. A particularly important case, where geometric regularization simplifies to a parametric (finite dimensional) manifold restriction is given by the Frobenius integrability theorem [10, 11]:

Theorem 1 (Frobenius theorem) *Suppose that C_f is an n-dimensional linear subspace of T_f. For any choice of $\phi(f) \in C_f$, the solution of Eq. (3) remains on an n-dimensional submanifold of $\mathcal{F}$ only, depending on the initial point f_0, iff C_f is involutive, i.e. for any two vector fields $\phi(f), \psi(f)$ in C_f we have that*

$$[\phi(f), \psi(f)] \in C_f,$$

where $[.,.]$ denotes a Lie bracket [11].

A simple example of an involutive regularization is when

$$C_f = \left\{ W_0 f + \sum_{k=1}^{r} W_k f^k + b \mid W_k \in \mathbb{R}^{m \times m}, b \in \mathbb{R}^m \right\}$$

where f^k are fixed functions. It is clear that the solution f_t remains in C_{f_0} from an initial f_0. Hence, this case corresponds to a linear regression. Selecting a nonlinear function $g : \mathbb{R} \to \mathbb{R}$, we can write a more general form of the geometric regularization discussed here, as follows,

$$C_f = \left\{ \Gamma(f) \left[W_0 f + \sum_{k=1}^{r} W_k f^k + b \right] \mid W_k \in \mathbb{R}^{d \times d_k}, b \in \mathbb{R}^d \right\} \tag{5}$$

where f^k are arbitrary fixed functions and $\Gamma(a)$ for $a = (a_1, a_2, \ldots, a_d) \in \mathbb{R}^d$ is a diagonal matrix with diagonals $\Gamma_{ii} = \mathrm{d}g/\mathrm{d}x(a_i)$.

3. ALGORITHMIC SOLUTION

The solution to the differential equation in Eq. (3) with the geometric regularization in Eq. (5) requires a specification of the tangent vectors $\phi(f) \in C_f$. To preserve a good control on the computations, and much like for the DNN architecture, we define $f : \mathbb{R}^n \to \mathbb{R}^d$, where the reduced dimension $d < n$ is a design parameter. Then, the desired function is calculated as $Df_\tau + c$ where $D \in \mathbb{R}^{m \times d}$ and $c \in \mathbb{R}^m$ are fixed. Letting $f_0(x) = Ux$ where $U \in \mathbb{R}^{d \times n}$ is a constant dimensionality reduction matrix, we rewrite the MSE objective in Eq. (1) as

$$L(f) = \mathbb{E}\left[\|\mathbf{y} - Df(\mathbf{x}) - c\|_2^2\right].$$

We subsequently apply the steepest descent principle to yield the following optimization:

$$\phi_f = \arg \min_{\phi \in C_f | \|\phi\|_2 \leq 1} \frac{\mathrm{d}L}{\mathrm{d}\tau}. \tag{6}$$

We next observe that under mild assumptions,

$$\frac{\mathrm{d}L}{\mathrm{d}\tau} = -\mathbb{E}\left[\left\langle \mathbf{z}\,,\; D\Gamma(f)\left[W_0 f + \sum_{k=1}^{r} W_k f^k + b\right]\right\rangle\right],$$

where $\mathbf{z} = \mathbf{y} - Df(\mathbf{x}) - c$ and W_0, W_k, b are to be decided based on the optimization in Eq. (6). After some manipulations, this leads to

$$\phi_f = \Gamma(f)\left[W_{0,f} f + \sum_{k=1}^{r} W_{k,f} f^k + b_f\right],$$

where

$$\begin{aligned} W_{0,f} &= \mathbb{E}\left[\Gamma(f(\mathbf{x}))D^T\mathbf{z}f^T(\mathbf{x})\right], \\ W_{k,f} &= \mathbb{E}\left[\Gamma(f(\mathbf{x}))D^T\mathbf{z}\left(f^k\right)^T(\mathbf{x})\right],\ k = 1, 2, \ldots, \\ b_f &= \mathbb{E}\left[\Gamma(f(\mathbf{x}))D^T\mathbf{z}\right] \end{aligned} \tag{7}$$

are specialized values of W_0, W_k, b, respectively.

3.1. Initialization

An efficient execution of the above procedure requires us to judiciously select the parameters U, D, c. We select U as the collection of basis vectors of the first d principal components of $\mathbf{x}$, i.e. $U = P_1^T$ where $\mathbb{E}[\mathbf{x}\mathbf{x}^T] = P\Sigma P^T$ is the Eigen-representation (SVD) of the correlation matrix, $P = [p_1\ p_2\ \ldots p_n]$ and $P_1 = [p_1\ p_2\ \ldots p_d]$. The matrices U, c are selected by minimizing the MSE objective with $f = f_0$. This yields,

$$\begin{aligned} D &= \mathbb{E}\left[\mathbf{y}f_0^T(\mathbf{x})\right]\mathbb{E}\left[f_0(\mathbf{x})f_0^T(\mathbf{x})\right]^{-1}, \\ c &= \mathbb{E}\left[\mathbf{y}\right] - D\mathbb{E}\left[f_0(\mathbf{x})\right]. \end{aligned}$$

This also affords us to update these matrices in the course of the optimization,

$$\begin{aligned} D &\leftarrow \mathbb{E}\left[\mathbf{y}f_t^T(\mathbf{x})\right]\mathbb{E}\left[f_t(\mathbf{x})f_t^T(\mathbf{x})\right]^{-1}, \\ c &\leftarrow \mathbb{E}\left[\mathbf{y}\right] - D\mathbb{E}\left[f_t(\mathbf{x})\right]. \end{aligned}$$

3.2. Momentum Method

Momentum methods are popular in machine learning and lead to considerable improvement in both performance and convergence speed [12, 13]. Since the originally formulated do not conform to our geometric regularization framework, we proposed an alternative approach to effectively mix the learning direction ϕ_f at each iteration with its preceding iterates to better control any associated rapid changes over iterations (low-pass filtering). Here to keep the geometric regularization structure, we instead mix the parameters W_k, b. This leads to the following modification in the original algorithm in Eq. (4):

$$\begin{aligned} f_{t+1} &= f_t + \mu_t\Gamma(f)\left[V_{0,t}f_t + \sum_{k=1}^{r} V_{k,t}f^k + e_t\right], \\ V_{k,t+1} &= \alpha_k V_{k,t} + W_{k,f_t},\ k = 0, 1, \ldots, r, \\ e_{t+1} &= \beta e_t + b_{f_t}, \end{aligned} \tag{8}$$

where W_{k,f_t}, b_{f_t} are given in Eq. (7).

3.3. Learning Parameter Selection

To select the remaining parameters α_k, β and μ_t, we manually tune parameters α_k, β, specifically utilize two strategies when tuning μ_t: a) fixing $\mu_t = \mu$ and b) using line search. The second method, we obtain by simple computations as

$$\mu_t = \frac{\mathbb{E}[\mathbf{z}_t^T D\psi_t]}{\mathbb{E}[\|D\psi_t\|_2^2]},$$

where $\mathbf{z}_t = \mathbf{y} - Df_t(\mathbf{x}) - c$, and

$$\psi_t = \Gamma(f_t)\left[V_{0,t}f_t + \sum_{k=1}^{r} V_{k,t}f^k + e_t\right].$$

3.4. Incorporating Shift Invariance

In the context of deep learning, especially for image processing, convolutional networks are popular. They differ from the regular deep networks in attempting to induce shift invariance in the linear operations of some layers, by way of convolution (Toeplitz matrix). We may adopt the same strategy in geometric regularization by further assuming that W_0 in Eq. (5) represents a convolution. We skip the derivations, for not only space limitation reasons, but also for their similarity to those leading to Eq. (7). The resulting algorithm with the momentum method is similar to Eq. (8) where $W_{0,f}$ is replaced by $W_{\mathrm{conv},f}$, defined as

$$W_{\mathrm{conv},f} = \arg\max_{W} \langle W, W_{0,f}\rangle,$$

where the optimization is over unit-norm convolution (Toeplitz) matrices. It turns out that since Toeplitz matrices form a vector space, $W_{\mathrm{conv},f}$ is a linear function of $W_{0,f}$ and can be quickly calculated [14]. Due to space limitation, we defer the details to [15].

4. THEORETICAL DISCUSSION

4.1. Relation to Deep Residual Networks

The proposed geometric regularization for nonlinear regression in Eq. (5) is inspired by the advances in the field of neural networks and deep learning. Recall that a generic deep artificial neural network (DNN) represents a sequence of functions (hidden layers) $f_0(x), f_1(x), \ldots, f_T(x)$ where $f_0(x) = x$ and f_t for $t = 0, 1, 2, \ldots$, is d_t-dimensional, where d_t is the network width in the t^{th} layer. The relation of these functions is plotted in Figure 1 (a). The so-called residual network (ResNet) architecture modifies DNNs by introducing bridging branches (edges) as shown by Figure 1 (b). We observe that the geometric regularization in Eq. (5) corresponds

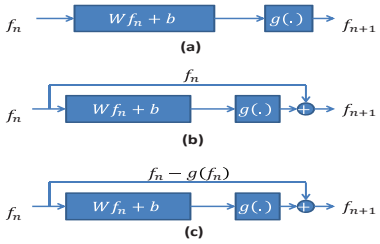

Fig. 1. Schematic scheme of a single layer in a)ANN b)ResNet and c)modified ResNet Architectures.

Method	Performance
plain iterations	97.4%
convolutional iterations	98.0%
2-stage learning	98.7%

Fig. 2. Performance of different learning strategies

to a modified version of ResNets, as depicted in Figure 1 (c), which can be written as

$$f_{t+1} = g(W_t f_t + b_t) - g(f_t) + f_t. \tag{9}$$

More concretely, when W_t, b_t are respectively near identity and near zero, i.e. $W_t = I + \epsilon \bar{W}_t$ and $b_t = \epsilon \bar{b}_t$ for small values of ϵ, we observe by Taylor expansion of Eq. (9) with respect to ϵ that the differential equation in Eq. (3) with geometric regularization in Eq. (5) provides the limit of the above modified ResNet architecture. This profound relation provides a novel approach for analyzing deep networks, which is deferred to [15].

5. NUMERICAL RESULTS

As a preliminary validation, we examine geometric regularization on the MNIST handwritten digits database including $50,000$ 28×28 black and white images of handwritten digits for training and $10,000$ more for testing [16, 17]. We note that state-of-the-art techniques already achieve an accuracy as high as 99.7%, thus justifying our validation study merely as a proof of concept. We use a single fixed function $f^1(x) = x$. In all experiments, we set $\alpha_0 = \beta = 0.98$ and let α_1 vary.

We have performed extensive numerical studies with different strategies (fixed or variable D, c, different step size selection methods and convolutional/plain layers), but can only focus on some key results due to space limitation. A more comprehensive comparison between these strategies is also insightful [15]. A summary of the best achieved performances is listed in Figure 2, where plain (non-convolutional) iterations are applied with fixed step size $\mu = 0.06$ and $\alpha_1 = 0.99$, $d = 400$ and fixed D, c. The convolutional iterations also include 2-D convolutional (Toeplitz) matrices with window

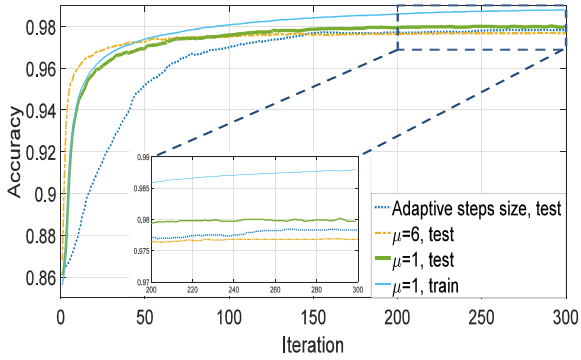

Fig. 3. Performance of different step size selection strategies.

length 5, fixed step size $\mu = 1$ and $\alpha_1 = 0.95$, and D, c updated at each iteration. We also consider a 2-stage procedure, where in the first 50 iterations, convolutional matrices are considered, and plain iterations are subsequently applied. In both stages, the step size is fixed to $\mu = 3$ and $\alpha = 0.95$, while the matrices D, c are updated at each iteration.

Figure 3 compares different strategies for step size selection with convolutional iterations by their associated performance, i.e. the fraction of correctly classified images in different iterations. The best asymptotic performance is obtained by fixing $\mu = 1$. Faster convergence may be obtained by larger step sizes at the expense of a decreased asymptotic performance. For example for $\mu = 6$, the algorithms reaches 96% accuracy in only 10 iterations and is 97% correct at 30. However, the process becomes substantially slower afterwards, which suggests a multi-stage procedure to boost performance. Using adaptive step size with line search shows a slightly degraded (higher than $\mu = 6$) performance, but dramatically decreases the convergence rate.

6. CONCLUSION

We proposed a supervised learning technique, which enjoys many common properties with deep learning, such as successive application of linear and non-linear operators, momentum method of implementation and convolutional layers. In contrast to deep learning, our method abandons the need for back propagation to hence improve the computational burden. Our method is semi-parametric as it essentially exploits a large number of weight parameters, yet avoiding over-parametrization. Another advantage of our technique is that it can theoretically be analyzed by tools in differential geometry as briefly discussed earlier. A conprehensive development is in [15]. The performance on the data sets we have thus far achieved, promises a great and unexplored potential waiting to be unveiled.

7. REFERENCES

[1] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, "Overview of supervised learning," in *The elements of statistical learning*, pp. 9–41. Springer, 2009.

[2] Simon S Haykin, *Neural networks: a comprehensive foundation*, Tsinghua University Press, 2001.

[3] Mohamad H Hassoun, *Fundamentals of artificial neural networks*, MIT press, 1995.

[4] G Gybenko, "Approximation by superposition of sigmoidal functions," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989.

[5] Kurt Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural networks*, vol. 4, no. 2, pp. 251–257, 1991.

[6] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] Michael Hauser and Asok Ray, "Principles of riemannian geometry in neural networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 2804–2813.

[9] Serge Lang, *Differential manifolds*, vol. 212, Springer, 1972.

[10] César Camacho and Alcides Lins Neto, *Geometric theory of foliations*, Springer Science & Business Media, 2013.

[11] Hassan K Khalil, "Noninear systems," *Prentice-Hall, New Jersey*, vol. 2, no. 5, pp. 5–1, 1996.

[12] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*, 2013, pp. 1139–1147.

[13] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[14] Albrecht Böttcher and Sergei M Grudsky, *Toeplitz matrices, asymptotic linear algebra, and functional analysis*, Birkhäuser, 2012.

[15] Ashkan Panahi and Hamid Krim, "a geometric view on representation power of deep networks," *under preparation*.

[16] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[17] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3642–3649.