A GENERAL APPROACH FOR LEARNING CONSTITUTIVE RELATIONSHIPS (PHYSICAL LAWS) FROM NEURAL NETWORKS

ANDREW TEMPLE, MATTHEW MILLER, AARON STEBNER, PETER COLLINS, AND BRANDEN KAPPES

1. Introduction

In the main, a disconnect exists between physically-based equations that describe observations of complex phenomena and models derived from machine learning methods, including neural networks. This disconnect results in misunderstandings (at a minimum) and philosophical gulfs (worse) between the data scientists and those scientists deeply rooted in disciplines which traditionally study the various complex phenomena. On the one hand, many academics and scientists have called machine learning, data analytics, and neural networks "black boxes", implying that they are either not understood or that they are simply another, perhaps crude, tool. On the other hand, there are those

2. Methods

Fully dense neural network (NN) architectures, such as the one shown in Figure 1, perform a sequence of affine transformations, $\mathbf{z}_i \leftarrow \boldsymbol{\theta}_i \mathbf{x}^{(i)}$, followed by element-wise functional operations, $\sigma(\mathbf{z}_i)$ to introduce non-linearity at each layer; that is, each layer stretches and distorts the underlying space.

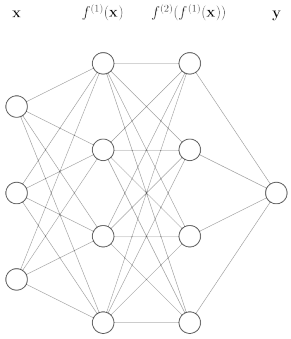

Figure 1. Schematic view of a fully dense neural network. Each sequence of affine and non-linear transformations are captured in the function, $f_i(\mathbf{x})$: $\mathbf{x}^{(i+1)} \leftarrow \sigma(\boldsymbol{\theta}_i \mathbf{x}^{(i)})$

The resulting network,

$$f(x) = \sigma(\boldsymbol{\theta}_n \sigma(\boldsymbol{\theta}_{n-1} \sigma(\dots \boldsymbol{\theta}_2 \sigma(\boldsymbol{\theta}_1 \mathbf{x})))) \tag{1}$$

is an arbitrary function generator, but at present, the network weights $\boldsymbol{\theta}_i$ can not map back to analytic forms that capture and describe the underlying physics. There are, however, many such mappings through polynomial series expansions,

$$f(x) = \sum_{n=0}^{\infty} a_n x^n \tag{2}$$

We hypothesize that the physics of a process can be extracted by fitting the polynomial expansions of known physical relationships to the polynomial coefficients of a polynomial series expansion of Equation (1).

The ReLU function is given below in Equation (3):

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$f(x) = \sigma(\theta_n \sigma(\theta_{n-1} \sigma(\dots \theta_2 \sigma(\theta_1 x)))) \tag{4}$$

The variable $\boldsymbol{x}$ represents a column vector containing $\boldsymbol{n}$ elements, and thus, the output from each node will be multplied by the respective weight, θ, before the application of the subsequent acitivation function.

e.g.,

$$\theta_1 \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \Rightarrow \begin{bmatrix} \theta_{11}x_1 + \theta_{12}x_2 + \dots \theta_{1n}x_n \\ \theta_{21}x_1 + \theta_{22}x_2 + \dots \theta_{2n}x_n \\ \vdots \\ \theta_{n1}x_1 + \theta_{n2}x_2 + \dots \theta_{nn}x_n \end{bmatrix} \tag{5}$$

Although ReLU (rectified linear units) have become a more common activation function, its discontinuity at $x = 0$ requires an infinite series to fully capture the behavior at this transition. However, the softplus function,

$$f(x) = log(1 + e^x) \tag{6}$$

The series expansion for the exponential

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} \tag{7}$$

where, $a_k = \frac{1}{k!}$, can be expressed as

$$e^x = \sum_{k=0}^{\infty} x^k a_k \tag{8}$$

A similar series expansion for the logarithm

$$log(1+x) = \sum_{n=1}^{\infty} (-1)^{n+1} \frac{x^n}{n} \tag{9}$$

where, $x = e^x$, allows for the softplus function to be represented as

$$f(x) = \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n} (\sum_{k=0}^{\infty} x^k a_k)^n \tag{10}$$

If the expansion of e^x is performed, then it can be seen that

$$(\sum_{k=0}^{\infty} x^k a_k)^n = (a_0 + x(a_1 + x(a_2 + xa_3...)))^n \tag{11}$$

Using the binomial theorem

$$(a+b)^n = \sum_{m=0}^{n} \binom{n}{m} a^{n-m} b^m \tag{12}$$

where, $a = a_0$ and $b = (x(a_1 + x(a_2 + xa_3...)))^n$, the series expansion of the exponential can be represented as

$$(\sum_{k=0}^{\infty} x^k a_k)^n = \sum_{m=0}^{n} \binom{n}{m} a_0^{n-m} (a_1 + x(a_2 + xa_3...))^n x^n \tag{13}$$

Continuing with this approach, it can be seen that ...

The analytical form, combining Equations (1) and (**??**), the estimated output of a two-layer NN can be written as an expansion:

$$
\begin{aligned}
\mathbf{y}_1 &= \sum_{k=0}^{\infty} a_k (\boldsymbol{\theta}_1^T \mathbf{x})^{\circ k} \\
\mathbf{y}_2 &= \sum_{k=0}^{\infty} b_k (\boldsymbol{\theta}_2^T \mathbf{y}_1)^{\circ k} \\
&= b_0 \mathbf{1} + \\
&\quad + b_1(\tilde{a}_0 + (\tilde{a}_1 + (\tilde{a}_2 + (\tilde{a}_3 + (\ldots)\tilde{\mathbf{x}})\tilde{\mathbf{x}})\tilde{\mathbf{x}})\tilde{\mathbf{x}}) \\
&\quad + b_2(\tilde{a}_0 + (\tilde{a}_1 + (\tilde{a}_2 + (\tilde{a}_3 + (\ldots)\tilde{\mathbf{x}})\tilde{\mathbf{x}})\tilde{\mathbf{x}})\tilde{\mathbf{x}})^2 \\
&\quad \vdots \\
&\quad + b_k(\tilde{a}_0 + (\tilde{a}_1 + (\tilde{a}_2 + (\tilde{a}_3 + (\ldots)\tilde{\mathbf{x}})\tilde{\mathbf{x}})\tilde{\mathbf{x}})\tilde{\mathbf{x}})^k \\
&\quad \vdots
\end{aligned} \tag{14}
$$

where $\tilde{a}_i = \boldsymbol{\theta}_2^T a_i$ and $\tilde{\mathbf{x}} = \boldsymbol{\theta}_1^T \mathbf{x}$. All $\boldsymbol{\theta}_i$, $\mathbf{x}$, and $\mathbf{y}$ are augmented to include the bias, $\mathbf{b}_i$, that is,

$$
\mathbf{x} \;:\; \mathbf{x} \leftarrow \begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix} = \begin{pmatrix} 1 \\ x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix} \tag{15}
$$

$$
\mathbf{y} \;:\; \mathbf{y} \leftarrow \begin{pmatrix} 1 \\ \mathbf{y} \end{pmatrix} \tag{16}
$$

$$
\boldsymbol{\theta}_i \;:\; \boldsymbol{\theta}_i \leftarrow \begin{pmatrix} \mathbf{b}_i & \boldsymbol{\theta}_i \end{pmatrix} \tag{17}
$$

The element-wise exponent operator, $\mathbf{x}^{\circ m} = (x_1^m \; x_2^m \; \cdots \; x_n^m)^T$, raises each element of a vector or matrix to the specified power, m.

From Equation (**??**), $a_i = 0$ for $i = 2, 4, 6, \ldots$, and therefore,

$$
\begin{aligned}
\mathbf{y}_1 =& \sum_{k=0}^{\infty} a_k (\boldsymbol{\theta}_1^T \mathbf{x})^{\circ k} \\
\mathbf{y}_2 =& \sum_{k=0}^{\infty} b_k (\boldsymbol{\theta}_2^T \mathbf{y}_1)^{\circ k} \\
=& b_0 \mathbf{1} + \\
&+ b_1(\tilde{a}_0 + (\tilde{a}_1 + (\tilde{a}_3 + (\tilde{a}_5 + (\ldots)\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}}) \\
&+ b_2(\tilde{a}_0 + (\tilde{a}_1 + (\tilde{a}_3 + (\tilde{a}_5 + (\ldots)\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}})^{\circ 2} \\
&\vdots \\
&+ b_k(\tilde{a}_0 + (\tilde{a}_1 + (\tilde{a}_3 + (\tilde{a}_5 + (\ldots)\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}}^{\circ 2})\tilde{\mathbf{x}})^{\circ k} \\
&\vdots \\
\mathbf{y}_2 =& \sum_{N=0}^{\infty} \sum_{k=0}^{N} \sum_{l=0}^{k} \sum_{m=0}^{l} \ldots b_N \binom{N}{k, l, m, \ldots} \tilde{a}_0^k \tilde{a}_1^l \tilde{a}_3^m \ldots \tilde{\mathbf{x}}^{\circ(N-k\ldots)} (\tilde{\mathbf{x}}^2)^{\circ(N-k-l\ldots)} (\tilde{\mathbf{x}}^2)^{\circ(N-k-l-m\ldots)} \\
&\qquad (18) \\
=& \sum_{N=0}^{\infty} \sum_{k=0}^{N} \sum_{l=0}^{k} \sum_{m=0}^{l} \ldots b_N \binom{N}{k, l, m, \ldots} \tilde{a}_0^k \tilde{a}_1^l \tilde{a}_3^m \ldots \tilde{\mathbf{x}}^{\circ(l+m+n+\ldots)} (\tilde{\mathbf{x}}^{\circ 2})^{\circ(m+n+\ldots)} (\tilde{\mathbf{x}}^{\circ 2})^{\circ(n+\ldots)}
\end{aligned}
$$

where $k + l + m + n + \ldots = N$. Collecting coefficients and terms of power k,

$$\mathbf{y}_2 = \sum_{k=0}^{\infty} c_k \tilde{\mathbf{x}}^{\circ k}$$

that, having the same form as Equation (**??**) creates a sequential process for determining the coefficients of the power series expansion of each layer in an ANN. Importantly, the output layer in a ANN regression is a single node with a linear activation, so the final layer, y_f, working from the last hidden layer, $\mathbf{y}_n$, is simply,

$$y_f = \boldsymbol{\theta}_n^T \mathbf{y}_n \tag{19}$$

3. Discussion

In the solution of the

To avoid implicit bias, data preprocessing is an important first step in training a neural network. Commonly, data is whitened, also known as scaling or standardizing, $\mathbf{x}_s : \mathbf{x}_s = \frac{\mathbf{x} - \overline{\mathbf{x}}}{\sigma}$, where $\overline{\mathbf{x}}$ is the arithmetic mean and σ is the standard deviation of the data set, $\mathbf{x}$. However, a model trained on such scaled data would change the space such that the primal space of the model (scaled) would differ from the primal space of the constitutive relationship (unscaled).

To accommodate whitened data, which is necessary to remove implicit bias while training a neural network, the constitutive relationship must also operate in the scaled primal space. After model training and fit of the physical constants in the constitutive relationship from $\boldsymbol{\alpha}$, the dual space of the neural network expansion, and $\boldsymbol{\beta}$, the dual space of the constitutive relationship expansion, the inverse transformation should be applied to closed-form constitutive relationship to extract the physical constant, e.g.

$$\begin{aligned} f(x) &= C_0 + C_1 x_s \\ &= C_0 + C_1 \left(\frac{x - \overline{x}}{\sigma} \right) \\ &= C_0' + C_1' x \end{aligned}$$

where

$$\begin{aligned} C_0' &= C_0 - C_1 \overline{x}/\sigma \\ C_1' &= C_1/\sigma \end{aligned}$$

Note that this accommodation may introduce a bias/offset even in constitutive relationships where no such offset was present before.

The Euclidean (L_2-norm) distance is used in optimizations, such as the training of neural networks and other machine learning algorithms, through the mean-square error,

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \|\mathbf{y} - \boldsymbol{\theta}\mathbf{x}.$$

However, because of the curse of dimensionality, the L_2-norm is not a desirable metric in high-dimensional spaces.

Many non-trivial problems in materials science, and in science more broadly, are explained not through a single constitutive relationship, but through a superposition of contributing physics.

Figure **??** shows an artificial dataset constructed to replicate the impact of yield stress in a two-phase, solid-solution strengthened alloy system. Using a combination of composite theory for the contribution of flow stress, NAME solid solution [**?**], and Hall-Petch [**?**] strengthening, the expected yield stress is

$$\sigma_y = F_v^A \sigma_f^A + F_v^B \sigma_f^B + \sum_i C_i [x_i]^{2/3} + \sum_j k_j d_j^{-1/2} + \dots \tag{20}$$

with free parameters

F_v^i	Volume fraction of phase i
$[x_i]$	Concentration of solute i
d_j	Average grain diameter of phase j

and fixed parameters

σ_f^i Flow stress of phase i
C_i Solid solution strengthening coefficient for solute species i
k_j Hall-Petch strengthening coefficient for phase j

The goal is to iteratively improve on this constitutive model one term at a time, and monitor the effect on the residuals between the predicted yield, $\hat{\sigma_y}$ and the actual yield σ_y.

Together, this leads to a seven-step process for systematically and incrementally extracting physics information from an ANN:

(1) Collect data–features and targets–for which relationships are expected to exist.
(2) Design and train a fully dense multi-layer perceptron network (ANN).
(3) Build a power series expansion from the architecture of this ANN, using Equations (**??**) and (18) to populate the coefficients using the trained weights from the neural network.
(4) Hypothesize a constitutive relationship between the feature space and the target space.
(5) Recast the terms in the hypothesis function from #4 as power series expansions, creating power series coefficient generating functions that are functions of the constitutive model fitting parameters. An example of this process is provided below, and a table of select power series expansions relevant to materials research are provided in Table (**??**).
(6) Perform an optimization, *e.g.* least squares, fit to find the fitting parameters from #5
(7) Calculate the residuals of the ANN power series expansion coefficient vector, and from this residual vector, the error in the model. If the accuracy is sufficient for the application, stop; otherwise, expand the constitutive relationship from step #4 and repeat.

TABLE 1. Examples of coefficient generating functions for functional forms commonly found in materials physics.

k	Ca^x	Cx^n	$Ce^{-\beta x}$	$Cx^{-1/2}$
0	1	–	C	C
1	$C\ln a$	–	$-\beta C$	$-\frac{1}{2}C$
2	$\frac{(\ln a)^2}{2}C$	–	$\frac{\beta^2}{2}C$	$\frac{3}{8}C$
⋮			⋮	
n	$\frac{(\ln a)^n}{n!}C$	$\begin{cases} C & \text{if } k=n \\ 0 & \text{otherwise} \end{cases}$	$(-1)^n\frac{\beta^n}{n!}C$	$C\prod_{i=1}^{n}(-1)\frac{2i-1}{2i}$