Parsimonious neural networks learn interpretable physical laws

Saaketh Desai\textsuperscript{1}, Alejandro Strachan\textsuperscript{2}

\textsuperscript{1}. Centre for Integrated Nanotechnologies, Sandia National Laboratories
\textsuperscript{2}. School of Materials Engineering, Purdue University
Machine learning models and their applications

Krizhevsky et al., Advances in neural information processing systems, (2012)

Machine learned models excel when data is plentiful
Encoding neural networks for genetic algorithms

Couple neural networks with genetic algorithms to balance interpretability and accuracy

Individual

$[1, 0, 1, 2, \ldots, 2]$

$act$: \{linear, squared, tanh, relu, \ldots\}

$\{W_{ij}\}$: \{0, 1, $\frac{1}{2}$, 2, \ldots, trainable\}

$F = f_1(E_{\text{test}}) + p \left( \Sigma_{i=1}^{n_a} w_i^2 + \Sigma_{j=1}^{n_w} f_2(w_j) \right)$

Keras

fitness  error on data  parsimony coefficient  simple activations  weight penalty

linear: 0  relu: 1  tanh: 2  0: 0  1, $\frac{1}{2}$, 2: 1  trainable: 2
How to train a PNN?

fitness based selection

crossovers/mutations

Fittest individual

Interpretable equation
Genetic operations on neural networks

Crossover

Mutation
Can we predict the melting temp based on fundamental inputs?

\[ T_{m}^{\text{lind}} = \left( \frac{4\pi^2}{9h^2} \right) f^2 a^2 m T_D^2 \]

Lindemann law developed in 1910

Poirier J., *Physics of the earth and planetary interiors* (1989)
Dimensional analysis on inputs

\[
\theta_0 = \frac{\hbar \nu_m}{k_b a} \quad \theta_1 = \frac{\hbar^2}{ma^2 k_b} \quad \theta_2 = \frac{a^3 G}{k_b} \quad \theta_3 = \frac{a^3 K}{k_b}
\]

Temperature units

\[
v_s = \sqrt{\frac{G}{\rho}} \quad v_p = \sqrt{\frac{K + \frac{4}{3} G}{\rho}}
\]

\[
v_m = \left[ \frac{3}{\left( \frac{1}{v_p} \right)^3 + 2 \left( \frac{1}{v_s} \right)^3} \right]^{\frac{1}{3}}
\]

Dimensionless inputs

\[
\theta_1' = \frac{\hbar}{mav_m} = \frac{\theta_1}{\theta_0}
\]

\[
\theta_2' = \frac{a^4 G}{\hbar \nu_m} = \frac{\theta_2}{\theta_0}
\]

\[
\theta_3' = \frac{a^4 K}{\hbar \nu_m} = \frac{\theta_3}{\theta_0}
\]

\[
\text{act: \{linear, multiply, squared, tanh, \ldots\}}
\]

\[
\{W_{ij}\}: \{0, 1, \ldots, \text{trainable}\}
\]
Launching the nanoHUB tool

Parsimonious neural networks

From your browser go to link: https://nanohub.org/tools/pnndemo/

Parsimonious neural networks

By Saaketh Desai, Alejandro Strachan

Design and train neural networks in conjunction with genetic algorithms to discover equations directly from data

Launch Tool

Version 1.1 - published on 05 Apr 2021
doi:10.21981/3Z9Z-4N07 cite this

Open source: license | download

View All Supporting Documents

Click on Launch Tool to begin
Discovering melting point laws

Melting temperature models

\[ T_{m}^{\text{PNN A}} = 21.8671 \theta_0 \]
\[ T_{m}^{\text{PNN B}} = 17.553 \theta_0 + 0.00198 \theta_2 \]
\[ T_{m}^{\text{PNN C}} = 11.903 \theta_0 + 0.0005 \theta_3 + 0.008 \theta_0^2 \theta_1 \]

Parsimonious neural networks learn non-linear interpretable laws
Discovering integration schemes from data

\[
x(t + \Delta t) = x(t) + 1.0001 \, v(t)\Delta t + 0.9997 \, \frac{1}{2} f \left( x(t) + v(t) \frac{\Delta t}{2} \right) \frac{\Delta t^2}{m}
\]

\[
v(t + \Delta t) = v(t) + 0.9997 f \left( x(t) + v(t) \frac{\Delta t}{2} \right) \frac{\Delta t}{m}
\]

Position Verlet integration scheme

Parsimonious neural networks learn underlying physics directly from data