Randomized function approximation

John Darges

Department of Mathematics
North Carolina State University

December 4, 2023

Learning vs. Approximation

Suppose we have a data set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^{d\times 1}$

Approximation

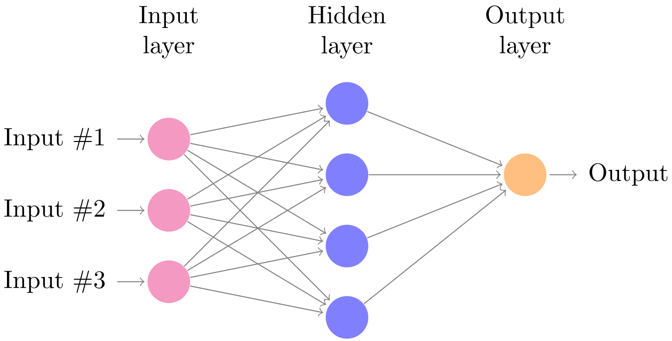
- Data comes from a known model $y_i = F(\boldsymbol{x}_i)$
- Inputs follow a known distribution $\boldsymbol{x} \sim \mathcal{K}$
- Can choose what is in our data set, but data may be expensive to generate

Learning

- Model is unknown but inputs/outputs are labeled
- Data follows some unknown distribution $(\boldsymbol{x}, y) \sim \mathcal{D}$
- Have access to some set of data (which is i.i.d.)

Feedforward neural networks

Artificial neural networks (ANNs) have very general structure (we focus on single layer neural networks)



Feedforward neural networks

Single layer NN has the form

$$F_{\mathrm{NN}}(\boldsymbol{x}) = \sum_{k=1}^{M} \alpha_k \sigma(W_j^\top \boldsymbol{x} + b_j) = \boldsymbol{\alpha}^\top \sigma(\mathbf{W}^\top \boldsymbol{x} + \boldsymbol{b}) \tag{1}$$

- $\mathbf{W} \in \mathbb{R}^{d\times M}$ is hidden layer weight matrix, $\mathbf{W} = \begin{bmatrix} W_1 & \dots & W_M \end{bmatrix}$
- $\boldsymbol{\alpha} \in \mathbb{R}^M$ outer weight vector
- $\boldsymbol{b} \in \mathbb{R}^M$ bias vector
- $\sigma : \mathbb{R} \to \mathbb{R}$ activation function (with universal approximation property[1])

To train, solve nonlinear least squares problem

$$\min_{\mathbf{W},\boldsymbol{\alpha},\boldsymbol{b}} \sum_{i=1}^{N} (F_{\mathrm{NN}}(\boldsymbol{x}_i; \mathbf{W}, \boldsymbol{\alpha}, \boldsymbol{b}) - y_i)^2 \tag{2}$$

[1] M. Leshno and V. Ya Lin and A. Pinkus and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. 1993.

Kernels

- Kernels are functions $K : \boldsymbol{X} \times \boldsymbol{X} \to \mathbb{R}$ (we will consider $\boldsymbol{X} = \mathbb{R}^d$)
- Some kernels are induced by a feature map $\phi : \mathbb{R}^d \to \mathcal{H}$
- $\mathcal{H}$ is a Hilbert space of certain functions $f : \mathbb{R}^d \to \mathbb{R}$
- Induced kernel is

$$K(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi_{\boldsymbol{x}}, \phi_{\boldsymbol{x}'} \rangle_{\mathcal{H}} \tag{3}$$

- Feature space $\mathcal{H}$ is a Reproducing kernel Hilbert space

$$f(\boldsymbol{x}) = \langle f, K(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}}, \quad f \in \mathcal{H}, \boldsymbol{x} \in \mathbb{R}^d \tag{4}$$

- Under right conditions, some function in RKHS can model our data

Kernel ridge regression

- Find $g \in \mathcal{H}$ so that $\langle g, \phi_{\boldsymbol{x}_i} \rangle_{\mathcal{H}} \approx y_i$, have a linear least squares problem

$$\min \sum_{i=1}^{N} (\langle g, \phi_{\boldsymbol{x}_i)} \rangle_{\mathcal{H}} - y_i)^2 = \sum_{i=1}^{N} (\langle f, K(\cdot, \boldsymbol{x}_i) \rangle_{\mathcal{H}} - y_i)^2 \tag{5}$$

- Construct kernel matrix $\mathbf{K} \in \mathbb{R}^{d \times d}$ where $\mathbf{K}_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, which is SPD
- Kernel ridge regression solves regularized least squares problem

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N} \left(\sum_{j=1}^{N} \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - y_i \right)^2 + \frac{\lambda^2}{2} \|\boldsymbol{\alpha}\|^2 \tag{6}$$

- Solution: $\boldsymbol{\alpha} = (\mathbf{K} + \lambda^2 I)^{-1} \boldsymbol{y}$ gives kernel machine

$$F_{\mathrm{KRR}}(\boldsymbol{x}) = \sum_{j=1}^{N} \alpha_j K(\boldsymbol{x}, \boldsymbol{x}_j) = \boldsymbol{\alpha}^{\top} K(\boldsymbol{x}) \tag{7}$$

Connection between Kernels and ANN

- Consider a feature map induced by an activation function

$$\phi_{\boldsymbol{\omega}}(\boldsymbol{x}) = \sigma(\boldsymbol{\omega}^\top \boldsymbol{x} + b) = \sigma(\boldsymbol{\omega}^\top (\boldsymbol{x}, 1)) \tag{8}$$

- We focus on Hilbert spaces with the L^2 inner product
- Functions in the RKHS look like

$$F(\boldsymbol{x}) = \int \alpha(\boldsymbol{\omega}) \sigma(\boldsymbol{\omega}^\top (\boldsymbol{x}, 1)) d\boldsymbol{\omega} \tag{9}$$

Randomization

- Randomized algorithms have become prevalent in numerical linear algebra
- Improve efficiency of smaller problems and feasibility of large problems
- Many ways to introduce to introduce randomness and still guarantee good results (almost surely)

Random weight neural networks

- Recall single layer NN

$$F_{\mathrm{NN}}(\boldsymbol{x}) = \sum_{k=1}^{M} \alpha_k \sigma(W_j^\top \boldsymbol{x} + b_j) = \boldsymbol{\alpha}^\top \sigma(\mathbf{W}^\top \boldsymbol{x} + \boldsymbol{b}) \tag{10}$$

- Instead of training over all $(d+2)M$ parameters, just randomly sample hidden layer weights and biases
- Only need to optimize output weights

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{N} (F_{\mathrm{NN}}(\boldsymbol{x}_i; \mathbf{W}', \boldsymbol{\alpha}, \boldsymbol{b}') - y_i)^2 = \|\mathbf{H}\boldsymbol{\alpha} - \boldsymbol{y}\|^2 \tag{11}$$

- Here $\mathbf{H}_{ij} = \sigma(W_j^\top \boldsymbol{x}_i + b_j)$
- Choices of activation function and sampling distribution matter!

History of random weight NNs

- Broomhead first introduces idea of turning training to linear least squares problem[2]
- Schmidt neural networks[3]
- Barron gives $\mathcal{O}(1/n^{2/d})$ convergence rates for sigmoidal networks [4]

[2] D.S. Broomhead and D. Lowe. Multivariable Functional Interpolation and Adaptive Networks. 1988.
[3] W.F. Schmidt, M.A. Kraaijveld, R.P.W. Duin. Feedforward neural networks with random weights. 1992.
[4] A. Barron. Universal Approximation Bounds for Superpositions of a Sigmoidal Function. 1993.

Random vector functional link

- Random vector functional link (RVFL)[5]

  Approximates functions with compact support, weights sampled uniformly

  Average asymptotic convergence and generalization bounds[6]

- Corrected theorems given in[7]

[5] Y.-H. Pao, G.-H. Park, D. Sobajic. Learning and generalization characteristics of the random vector functional-link net. 1994.

[6] B. Igelnik, and Y.-H. Pao. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. 1995.

[7] D. Needell, A. Nelson, R. Saab, P. Salanevich. Random Vector Functional Link Networks for Function Approximation on Manifolds. 2022.

Extreme learning machine

- Extreme learning machine (ELM) includes neural network and radial basis function versions[8]
- Prove universal approximation when activation function is bounded and sampling distribution is continuous[9]
- Claim much more broad approximation capabilities[10]
- No convergence or generalization guarantees

[8] G.-B. Huang and D. Wang and Y. Lan. Extreme learning machines: A review. 2011.

[9] G.-B. Huang, L. Chen, and C.-K. Siew. Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes. 2006.

[10] G.-B. Huang, L. Chen. Convex incremental extreme learning machine. 2007

Kernel methods the random feature way

- Consider a kernel $K(\boldsymbol{x}, \boldsymbol{x'}) = \int \phi_{\boldsymbol{x}}(\boldsymbol{\omega})\phi_{\boldsymbol{x'}}(\boldsymbol{\omega})d\boldsymbol{\omega}$
- For a large data set, impractical to compute (and store) full kernel matrix $\mathbf{K}$.
- Instead approximate kernel matrix by a rank one approximation
- Random features[11] use Monte Carlo sampling

$$\mathbf{K} \approx \sum_{k=1}^{K} \boldsymbol{z}_k \boldsymbol{z}_k^\top, \quad \boldsymbol{z}_k = [\ \phi_{\boldsymbol{x}_1}(\boldsymbol{\omega}_k) \quad \ldots \phi_{\boldsymbol{x}_N}(\boldsymbol{\omega}_k)\ ]^\top \tag{12}$$

- $\mathcal{O}(\sqrt{n}\log(n))$ features give $\mathcal{O}(1/\sqrt{n})$ bounds [12]

[11] A. Rahimi, B. Recht. Random Features for Large-Scale Kernel Machines. 2007.
[12] A. Rudi, L. Rosasco. Generalization Properties of Learning with Random Features. 2017.

Random bases

- Random basis expansion[13] does not work with the kernel, instead the feature map
- With neural network style feature maps, they are equivalent to random weight neural networks
- Can take advantage of RKHS theory and functional analysis[14]
- RKHS should be dense in space of continuous functions
- We should be able to express the following by a series

$$\int f(\omega)\phi_{\mathbf{x}}(\omega)d\omega \tag{13}$$

- Can recreate claim of Huang '06 using function analysis[15]

[13] A. Rahimi, B. Recht. Uniform approximation of functions with random bases. 2008.
[14] F. Bach. On the Equivalence between Kernel Quadrature Rules and Random Feature Expansions. 2017.
[15] Y. Sun, A. Gilbert, A. Tewari. On the Approximation Properties of Random ReLU Features. 2019.

Random bases with structure

- Do random bases/neural networks have any advantages over random features?
- We have broad choices for activation functions
- Sampling distribution for $\boldsymbol{\omega}$ has many choices, too
- By clever sampling, can impose function structure (interactions/main effects) on

$$F(\boldsymbol{x}) = \sum_{k=1}^{M} \alpha_k \sigma(W_j^\top \boldsymbol{x} + b_j) \tag{14}$$

- In[16] and[17] use sparse sampling ($W \sim X \cdot Y$, X continuous RV and Y Bernoulli RV) to impose structure

[16] A. Hashemi, H. Schaeffer, R. Shi, U. Topcu, G. Tran, R. Ward. Generalization bounds for sparse random feature expansions. 2023.

[17] J. Darges, A. Alexanderian, P. Gremaud. Extreme learning machines for variance-based global sensitivity analysis. 2023.

References

1. M. Leshno and V. Ya Lin and A. Pinkus and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. 1993.
2. C.E. Rasmussen, C.K.I. Williams. Gaussian Processes for Machine Learning. 2006.
3. R. Schaback, H. Wendland. Kernel techniques: From machine learning to meshless methods. 2006.
4. P. Martinsson, J.A. Tropp. Randomized numerical linear algebra: Foundations and algorithms 2020.
5. D.S. Broomhead and D. Lowe. Multivariable Functional Interpolation and Adaptive Networks. 1988.
6. W.F. Schmidt, M.A. Kraaijveld, R.P.W. Duin. Feedforward neural networks with random weights. 1992.
7. A. Barron. Universal Approximation Bounds for Superpositions of a Sigmoidal Function. 1993.

References

1. Y.-H. Pao, G.-H. Park, D. Sobajic. Learning and generalization characteristics of the random vector functional-link net. 1994.
2. B. Igelnik, and Y.-H. Pao. Stochastic choice of basis functions in adaptive function approximation and the functional-link net. 1995.
3. D. Needell, A. Nelson, R. Saab, P. Salanevich. Random Vector Functional Link Networks for Function Approximation on Manifolds. 2022.
4. G.-B. Huang, L. Chen, and C.-K. Siew. Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes. 2006.
5. G.-B. Huang and D. Wang and Y. Lan. Extreme learning machines: A review. 2011.
6. A. Rahimi, B. Recht. Random Features for Large-Scale Kernel Machines. 2007.
7. A. Rahimi, B. Recht. Uniform approximation of functions with random bases. 2008.
8. F. Bach. On the Equivalence between Kernel Quadrature Rules and Random Feature Expansions. 2017.

References

1. Y. Sun, A. Gilbert, A. Tewari. On the Approximation Properties of Random ReLU Features. 2019.
2. A. Rudi, L. Rosasco. Generalization Properties of Learning with Random Features. 2017.
3. A. Hashemi, H. Schaeffer, R. Shi, U. Topcu, G. Tran, R. Ward. Generalization bounds for sparse random feature expansions. 2023.
4. J. Darges, A. Alexanderian, P. Gremaud. Extreme learning machines for variance-based global sensitivity analysis. 2023.
5. F. Liu and X. Huang and Y. Chen, J. K. Suykens. Random Features for Kernel Approximation: A Survey on Algorithms, Theory, and Beyond. 2022.
6. M. Nguyen, N. Mücke. Random feature approximation for general spectral methods. 2023.
7. P.N. Suganthan, R. Katuwal. On the origins of randomization-based feedforward neural networks. 2021.
8. W. Cao, X. Wang, Z. Ming, J. Gao. A review on neural networks with random weights. 2018.