Mathematical Foundations of Machine Learning – Spring 2019

Project list

Theoretical projects

1. [Anisotropic Besov smoothness] Let $f(x)=\mathrm{I}_{\tilde{\Omega}}(x)$, where $\tilde{\Omega}\subset[0,1]^n$ is convex with smooth boundary. Show that
   a. For a tree $\mathcal{T}_D$, created via non-adaptive partition along dyadic lines, we have $f\in B_\tau^\alpha(\mathcal{T}_D)$, for $\alpha<1/n\tau$.
   b. For $n=2$, show there exists an adaptive anisotropic tree $\mathcal{T}_A$, such that $f\in B_\tau^\alpha(\mathcal{T}_A)$, for $\alpha<2/3\tau$.

2. [Scattering Networks] Scattering Networks are a special (simplified) form of convolutional neural networks where the coefficients of the filters are predetermined and not learnt. Provide an overview of the method [6] and in particular, a sketch of the "stability to deformation" property [5].

Applied projects

General comments

A. Use publically available datasets such as the UCI machine learning repository (http://archive.ics.uci.edu/ml/index.php)
B. In all experiments use 5 fold cross validation.
C. For regression problems provide average error and std of error statistics.
D. For classification problems provide accuracy (TP+TN)/(P+N), precision TP/(TP+FP) and recall TP/(TP+FN) statistics.
E. Perform hyper-parameter search and try to explain the logic of the best configuration.
F. "Debug" your results: look at confusion matrices, investigate your false positive and negatives. Try to understand where your models fail and try to fix them.
G. For ML problems, compare your results to the results using standard models from Scikit-Learn, R, etc.
H. Try to come up with other ideas beyond the basic project description.

3. [Feature importance via wavelet decomposition of RF] Reproduce the feature importance results of [1]
   a. Provide summary of the wavelet-based method with emphasis on the use of the validation set to determine a threshold for wavelet norms.
   b. Test on regression & classification problems (multi-class problems).
   c. Observe differences (if any) on small/large datasets.
   d. Compare extensively with standard methods as in [1]

4. [Compression & denoising with wavelet decomposition of RF] Reproduce and add to the results of [1]

a. Compression – Investigate the RF compression capabilities of wavelets through tradeoff between the number of trees and tree components versus the prediction error.
b. Denoising – add various levels of Gaussian noise to regression datasets and add various levels of mis-labeling to classification datasets. Investigate the performance of wavelet denoising.

5. [Function space analysis of DL] Reproduce the research of the paper [3]
    a. Train 'small' TensorFlow networks for MNIST, CIFAR10 datasets.
    b. Perform Besov smoothness analysis of the representation layers at various stages of the training
    c. Try also to experiment with different network configurations and investigate the relationship between the classification error on the testing set and the smoothness analysis of the representation layers.

6. [Function space analysis of ResNets] Add to the research of [3]
    a. Train a 'small' version of a ResNet [4] on the CIFAR10 dataset.
    b. Investigate the performance and perform Besov smoothness analysis of the network with and without the residual connections.

7. [Function space analysis of Transfer learning] Add to the research of [3]
    a. Train a network on the MNIST dataset.
    b. Perform Besov smoothness analysis.
    c. Apply transfer learning on a 'small' set of (grayscale) CIFAR10 using as basis a network from (a). This implies 'freezing' some of the first layers and re-training the last layers or creating and training new last layers.
    d. Perform Besov smoothness analysis of the transfer-learning architecture using the full CIFAR10 set.

8. [Deep neural decision forest] Follow [7] to create an architecture that combines CNN architecture with differentiable decision tree/forest
    a. Use MNIST and CIFAR10 datasets.
    b. Compare to the 'standard' architecture with the softmax layer.

9. [Scattering Networks for small datasets] Use the methods of [6] to compare the results of trained convolutional neural networks over Scattering Networks in the cases of small/simple datasets.

    a. Use the MNIST dataset as an example for a 'simple' dataset.
    b. Use various sizes of the CIFAR10 dataset.

10. [AI for numerical PDEs] Follow [8] to apply a DL solution to a PDE
    a. Use example 3.1.1 of [8] as a base for your experiments.
    b. Try examples where you know what the analytic solution is
    c. Try to compare with other 'off-the-shelf' solvers.

11. [AI for numerical PDEs] Apply a DL approach to inverse problems of the wave equation

a. Create via simulations a dataset of waves with time [0,1000] and different source locations. Use as domain a square with a grid of 128x128.
b. Train a regression DL network on dataset of images at time 1000 to predict source location.
c. Train a regression DL network on dataset of images at various times [500,1000] to predict source location.

References

[1] O. Elisha and S. Dekel, Wavelet decompositions of Random Forests - smoothness analysis, sparse approximation and applications, JMLR 17 (2016).

[2] O. Morgan, O. Elisha and S. Dekel, Wavelet decomposition of Gradient Boosting, preprint.

[3] O. Elisha and S. Dekel, Function space analysis of deep learning representation layers, preprint.

[4] H. Kaiming, Z. Xiangyu, R. Shaoqing and SD Jian, Residual Learning for Image Recognition, proceedings of CVPR 2016.

[5] S. Mallat, Group Invariant Scattering, Comm. Pure and Applied Math 65 (2012), 1331-1398.

[6] J. Bruna and S. Mallat, Invariant Scattering Convolution Networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (2013), 1872 – 1886.

[7] P. Kontschieder, M. Fiterau, A. Criminisi and S. Rota Bul`o, Deep Neural Decision Forests, ICCV 2015.

[8] M. Raissia, P. Perdikarisb and G.E.Karniadakisa, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics 378(2019), 686–707.