Complex systems exhibit structural hierarchies, and efficient models of those systems must reflect that hierarchy. For example it is not efficient to model mechanical systems at the atomic level or images at the pixel level. Each of these systems exhibits many layers of structural and behavioral hierarchy built atop these primitives and practical models of those systems must be matched to that hierarchy. In the same way that implementation of the computation for a multiplier with one layer of logic is enormously larger than implementation of a multiplier with many layers of logic, deep models are more efficient than shallow models.

## Motivating Themes

* Hierarchy in natural problems
* Depth for computational economy in hierarchical problems
* Decimation to reflect temporal hierarchy
* Regularization through dropout
* Regularization through invertible representations
* Regularization through sparsity

## Learning Objectives

* Develop basic python programming skills.
* Apply an existing machine learning library with GPU acceleration to duplicate a published result.
* Apply a neural network to predict time series data.
* Apply a neural network to classify time series data.
* Implement a reference time series learning algorithm.
* Implement an experimental time series learning algorithm.
* Contrasting performance of the experimental algorithm with the reference.

## First Steps

* Gather standard benchmark datasets from their owners to preserve pedigree. (MNIST, CIFAR, …)
* Develop a standard interface to encapsulate the datasets. (raw, preprocessed, labeled, tagged, etc…)
* Select an existing library to provide GPU support for neural network computation. The three major deep learning labs have developed various libraries to support their work. Professor Wolfe suggested that using the simple cudamatrix library may be most appropriate given that I would like to implement new techniques rather than simply using existing techniques. For this application, the power and sophistication of a complex library may prove more prohibitive rather than enabling.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Package | Lab | Language | Description | Dependencies |
| Torch7 | LeCun | Lua | Machine learning framework with packages for neural networks. |  |
| cudamatrix | Hinton | Python | Matrix class similar to numpy.ndarray. | numpy, nose |
| cuda-convnet | Hinton | Python | Convolutional neural network class |  |
| deepnet | Hinton | Python | Multiple deep learning algorithms (RBM, convnet, etc.) | Cudamatrix, Cuda-convnet |
| Theano | Bengio | Python | Tensor class, symbolic gradient computation | Python, numpy, scipy, BLAS, g++ |
| Pylearn2 | Bengio | Python | Neural network framework | Theano, PyYAML, PIL |

* Use the library to duplicate (approximately) one or more published neural network performance results in order to establish correct operation of the algorithms.
* Conduct a literature survey to identify competing approaches for time series classification or time series prediction problems. The scope should not be limited to neural networks and might uncover useful learning problems, datasets, and algorithms to serve as comparison benchmarks. Professor Wolfe suggested this as an early task to avoid wasting effort reinventing the wheel.
* Prepare one or more time series datasets. (Temperature, Speech, Price)

## Deep Learning Thought Leaders

|  |  |  |
| --- | --- | --- |
| Lab | University Affiliation | Corporate Affiliation |
| Geoffrey Hinton | University of Toronto | Google Research |
| Yoshua Bengio | University of Montreal |  |
| Yann LeCun | NYU | Facebook |
| Andrew Ng | Stanford | Baidu |
| Juergen Schmidhuber | IDSIA |  |
| Christopher Bishop | University of Edinburg | Microsoft Research |
| David Mackay | University of Cambridge |  |