Introduction to Deep Learning

Mnacho Echenim

Grenoble INP-Ensimag

2022-2023

Outline of the project

- Goal: construct a neural network (almost) from scratch in C#
  - The MathNet library (Nuget) is available for matrix computations
  - Some source code is provided:
    - A class library with interfaces for the neural networks
    - Several applications to help training and evaluating the implemented networks
- Use the neural network to solve a regression problem
  - The neural network will be used to price a financial product
  - Competition to select the best neural network
- Evaluation
  - The source code
    - Correctness of the implementation
    - Structure, maintainability, efficiency
    - Implemented features
  - (How well the best network performs)
    - The serialized version of the best network that was constructed

Global timetable

- Hands-on 1: Forward propagation (1.5 hours)
  - Update of the code to make forward propagation work
  - Validation: test network proxies for boolean functions
  - Refactoring of the code to reduce the number of instantiations
- Hands-on 2: Backpropagation, gradient descent (1.5 hours)
  - Update of the layer deserializer to retrieve gradient adjustment parameter
  - Update of the code to make backpropagation and gradient descent work
  - WPF and console apps for training the networks are provided
  - Implement a network serializer to save the trained networks
- Hands-on 3: Mini-batch, gradient acceleration (4.5 hours)
  - Update of the code to include batch size
  - Implementation of Momentum gradient acceleration
- Hands-on 4: Regularization (1.5 hours)
  - Implementation of an L^2 layer
- **Pricing neural network** (6 hours)

Available resources

- A solution with the following readonly projects
  - BooleanFunctionTester
  - BooleanFunctionTesterConsole
  - DataProviders
  - NeuralNetwork.Common
  - PropagationComparison
  - RegressionConsole
  - Trainer
  - TrainingConsole
- The skeleton of a project to modify
  - NeuralNetwork
- A description of the projects
- Quickstart examples for the projects that generate an executable file

NeuralNetwork.Common

Common architecture for the neural networks that will be constructed

- `INetwork.cs`: interface describing a neural network
- `IComponentWithMode.cs`: interface that permits having a network that switches from a training to an evaluation mode (e.g. Dropout)
- `MathData.cs`: container class for the data used to train or evaluate a network
- **Layers**: contains an interface that should be implemented by all layers, and an enum of layer types
- **Activators**: implementations of some activators that can be used in the layers
- **GradientAdjustmentParameters**: container classes for the parameters of some gradient optimization methods
- **Serialization**: container classes of the information to be stored when persisting a network
- **JsonUtils**: utility classes to read and write persisted networks in Json format

NeuralNetwork, DataProviders

- `Network.cs`: initial implementation of a neural network. To be completed and modified
- **Layers**: contains an initial implementation of a standard layer, to be completed and modified. Other layer implementations will go in this folder
- **Serialization**: contains initial implementations of serialization/deserialization utility classes. To be completed
- **DataProviders project**: contains providers to retrieve pricing data, as well as data for the `and` and `xor` boolean functions

Forward and backpropagation

- Two projects are used to test that forward propagation has been correctly implemented:
  - **BooleanFunctionTesterConsole**
  - **BooleanFunctionTester** (Windows only)
- Three projects are used to train neural networks:
  - **Trainer**: core project for training a network
  - **TrainingConsole**: project to invoke the `Trainer` from a console application
  - **Visualizer** (Windows only): UI for training a network

Validation

- **PropagationComparison**: console application that performs:
  - one step of forward propagation
  - one step of backpropagation
  - one last step of forward propagation

  The results of both forward propagation steps are output to the console or a file
- **RegressionConsole**: console application that outputs a statistics summary of a network performance on pricing data