# System Requirements Specification

**EasyNN/EasyAI**

**CS 491, Spring 2021**

Team Name: EasyNN/EasyAI

Team Members:

- Jack Nguyen
- David Fadini
- Liam Kehoe
- Nathan Rose
- Nathan Foster

Contents of this Document

---

**Section 1: Introduction**

System to be Produced:

- EasyAI: This product is designed to allow users to create or log in to an account, allowing them access to the website and letting users access frameworks in the website. These frameworks will include the EasyNN framework and the EasyGA framework. The users will also be able to adjust the framework's attributes from a form on the website. On the website, users will be able to save their work to a database allowing access to their data at a later time.

- EasyNN: The goal of this product is to create a neural network framework that is easy to use and customize. This will be done in python and heavily leveraging default values for the user to start training immediately and then dive into various features as they need to.

Applicable Standards

- <You do not have to repeat the standards included in the project plan. Instead, cite any standards that are specific to the system requirements.>

Definitions, Acronyms, and Abbreviations

- <Include any that are needed to read this document or "none" if the document is self-explanatory and no acronyms or abbreviations will be used

- A model is a representation of another system.

- A (Artificial) Neural Network (NN or ANN) is a simplified computational model of the human brain, loosely based on the idea of passing information between several neurons to get an output.

- A Neuron is a component in a Neural Network that collects, stores, and sends values.

- A loss function is a numerical measurement of the performance of a model.

- A topology is a structure defined by an arrangement of nodes and their connections.

- Optimization is the process of either minimizing or maximizing a loss function.

- An Optimizer is something that performs optimization.

- Neuroevolution is a class of general Neural Network optimization techniques for optimizing topologies based on Genetic Algorithms.

- Machine learning is the process of optimizing a model.

- Forward Propagation is the process of processing values through a Neural Network, starting from the input nodes and moving to the output nodes.

- Back Propagation is the process of computing components of the gradient iteratively instead of entirely at once, starting from the output nodes and moving to the input nodes.

- A tensor is a multidimensional array with a shape and indexing by references/pointers. Examples of tensors are floats (except for the indexing), vectors, and matrices.

- A multidimensional array's shape is the list of the lengths of each of its dimensions. If a dimension does not have consistent lengths, then the array does not have a shape.

- Indexing by references/pointers means that if one gets a part of the tensor and modifies it, it should modify the original tensor.

**Section 2: Product Overview**

Assumptions:

- <List all the assumptions the developers are making. For example assumptions about other systems this product will interface with; assumptions about the technological environment in which the product will operate (how much memory, what type of processor, ...); assumptions about availability and capability of COTS, GOTS, or other re-used products, ...>

- EasyGA:

    - The system shall attain better fitness values over time.

- EasyAI:

    - The system can accept new and returning users

    - The system can store users data in a database

    - The system is written in python using the Django framework

Stakeholders:

- <A stakeholder is anyone who has an interest in the system to be developed. For example, the customer, the various classes of users, applicable regulatory agencies, ... List each category of stakeholder and give a phrase or a sentence to describe their interest or concerns>

- EasyAI:

    - Users: This person wants a fully developed system to allow them to run projects through the neural network frameworks.

- Daniel, Product Owner: Wants the website to run the EasyNN and EasyGA and provide an interface for users to run their problems.

- EasyNN:
  - User: The person who wants to be able to set up and customize the neural network to solve their problem.
  - Daniel, Product Owner: Wants the system developed to build on top of his EASYGA system.

Event Table:

**Easy NN**

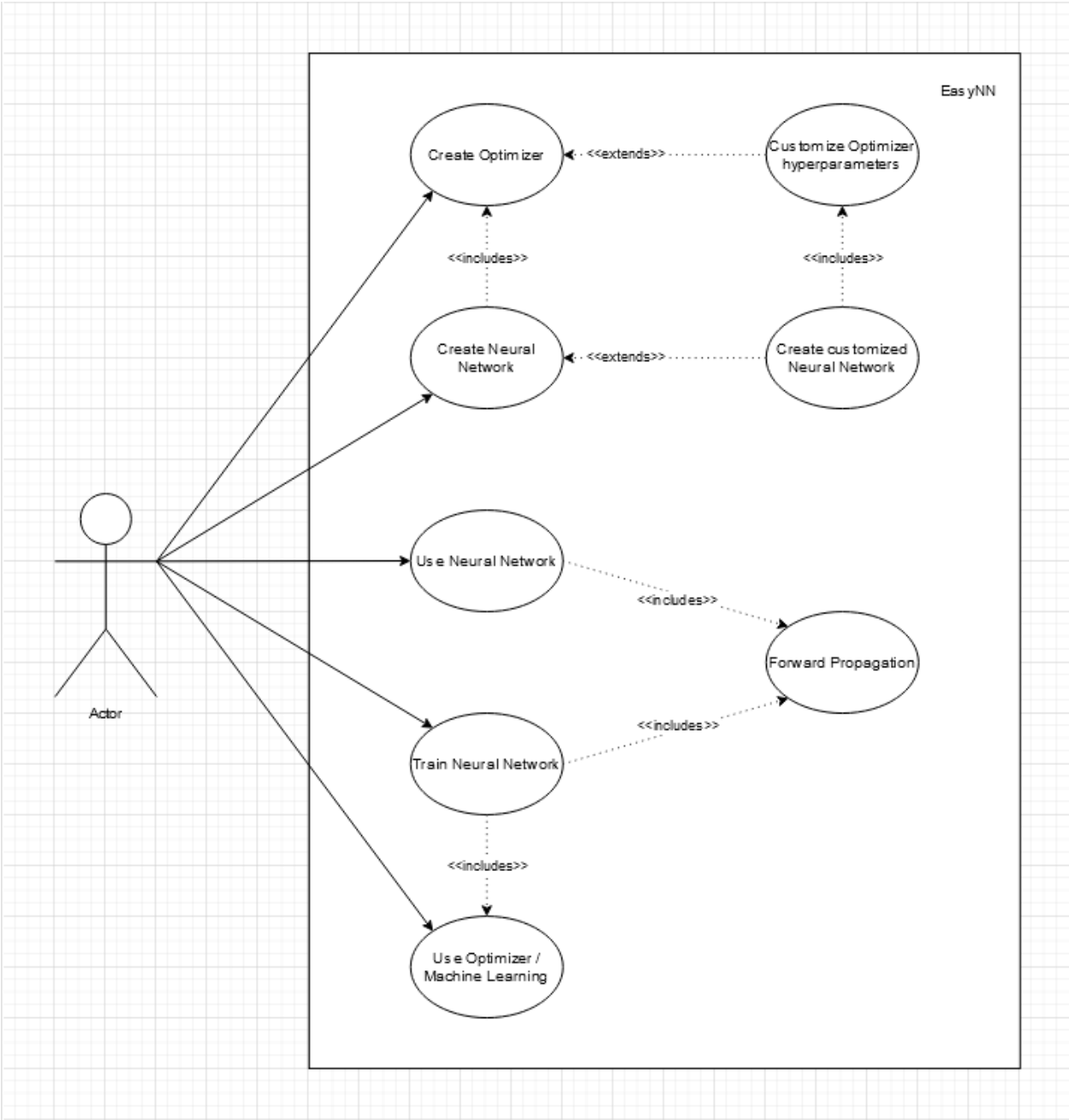
| Event Name | External Stimuli | External Responses | Internal data and state |
|---|---|---|---|
| Creation | Constructor is called | A neural network is created | by default it is set up for good training of [[Whatever the database is called]] |
| Forward Propagation | Data is given to NN to process | The confidence interval is given | The values for each layer are determined(and stored) |
| Backward Propagation | The correct answer is given to NN | loss and accuracy are given for the last input/batch | All weights and biases are updated to improve the accuracy of the model |
| | | | |

**Easy AI**

| Event Name | External Stimuli | External Responses | Internal data and state |
|---|---|---|---|
| Create User | A database is filed | A user account is created | The values of the user ID, name, password [other credentials are changed] |
| Log In | A function with parameters are passed | The user profile is loaded into the system | The profile will load the user's tables of different projects |
| Algorithm Configuration | A form is filled with configuration parameters | The algorithm loads the selected parameters | the default will change parameters and become what the user wants |

**EasyNN**

Use Case Diagram



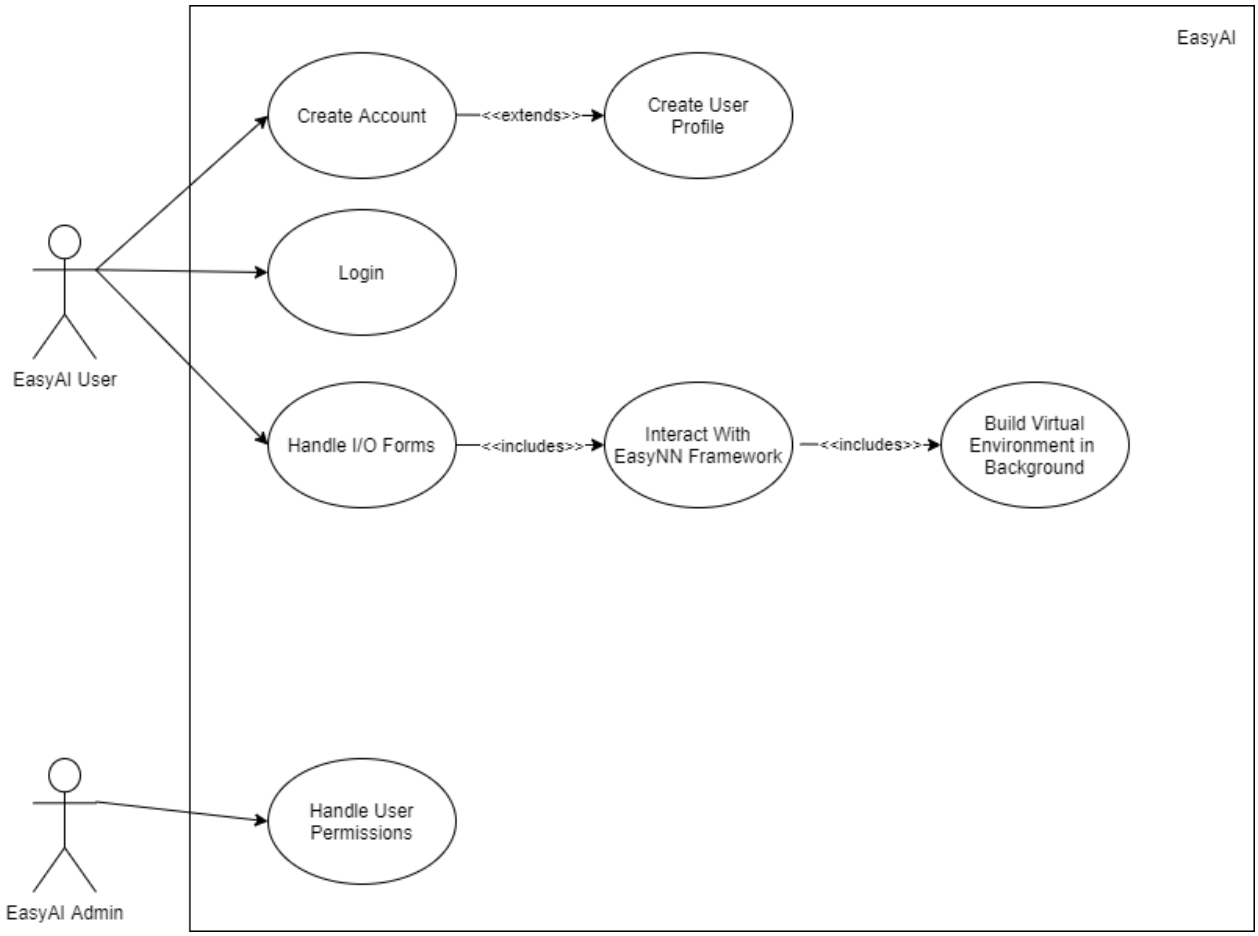

Use Case Descriptions:

1. Create Optimizer
    a. The user creates an Optimizer to optimize a function.
2. Customize Optimizer hyperparameters
    a. The user tweaks parameters in the Optimizer to improve performance for their problem.

3. Create Neural Network
    a. The user creates a basic Neural Network that will work with the MNIST dataset.
4. Create Customized Neural Network
    a. The user customizes their Neural Network for their problem.
5. Use Neural Network
    a. The User can feed data through a Neural Network.
6. Train Neural Network
    a. The user can give the Neural Network training data to adjust to.
7. Forward Propagation
    a. The part of the Neural Network that determines what the output is given a set of inputs.
8. Use Optimizer / Machine Learning
    a. The user optimizes a function through the Machine Learning system.

**EasyAI**

Use Case Diagram




Use Case Descriptions:

1. Create Account

    a. The user can register an account with the website
2. Create User Profile
    a. A user profile is made for the user after they register an account
3. Login
    a. A user who has already registered for an account can log in to their account
4. Handle I/O Forms
    a. The EasyAI application can handle input and output forms that the user will fill out
5. Interact With EasyNN Framework
    a. The EasyAI application can interact with the EasyNN framework
6. Build Virtual Environment in Background
    a. The EasyAI application can create a virtual environment in the background so that users can input code without corrupting the site
7. Handle User Permissions
    a. An admin can log in to the admin section and handle user permissions

**Section 3: Specific Requirements**

<Use the following template for <u>each</u> requirement. >

| |
|---|
| No: <unique requirement number> |
| Statement: <the "shall" statement of the requirement> |
| Source: <source of the requirement> |
| Dependency: <list each other requirements on which satisfaction of this requirement depends. (Maybe "None")> |
| Conflicts: <list each other requirements with which this requirement conflicts. (Maybe "None")> |
| Supporting Materials: <list any supporting diagrams, lists, memos, etc.> |
| Evaluation Method: <How can you tell if the completed system satisfies this requirement? > |
| Revision History: <who, when, what> |

3.1 Functional Requirements

- < Describe the fundamental <u>actions</u> that the system must perform. Functional requirements can be partitioned into subfunctions or subprocesses. Note: the System design partition does not have to correspond with the functional requirements partition. Functional requirements include:
    - validity checks on the inputs,

- the exact sequence of operations,
- responses to abnormal situations
- relationship of outputs to inputs
    - input/output sequences, formulas for input to output conversion, etc.
- ...>

3.1.1 EasyNN

3.1.1.1
Description: The Machine Learning system shall find the local minima of a function, provided a built-in optimization algorithm.
Evaluation: The Machine learning system should be tried on test cases from https://en.wikipedia.org/wiki/Test_functions_for_optimization.

3.1.1.2
Description: The Optimizer shall use given values and derivatives to train on.
Evaluation: The Machine Learning system should pass the values and derivatives into the optimizer.

3.1.1.3
Description: The EasyNN system shall save network models in a database when asked to.
Evaluation: The EasyNN system should save the topology, machine learning parameters, and activation functions to a database.

3.1.1.4
Description: The EasyNN system shall have all neural network models as listed in 4.1.
Evaluation: Implementations of each model should be included.

3.1.1.5
Description: The EasyNN system shall have all optimization algorithms as listed in 4.2.
Evaluation: Implementations should follow formulas found in the literature.

3.1.1.6
Description: The Neural Network shall use backpropagation (through time) to compute derivatives during training. Recurrent models require a modified form of back propagation.
Evaluation: Implementations of back propagation should follow formulas found in the literature.

3.1.2 EasyAI

3.1.2.1
Description: The EasyAI web application shall have a register page to allow users to have their page.
Evaluation: Created account information shall be saved and shown on the admin page.

3.1.2.2
Description: The EasyAI web application shall have a secure login page.
Evaluation: Entered account information should be checked against the existing user information database and the user shall be logged in if credentials match.

3.1.2.3
Description: The EasyAI web application shall interact with the EasyNN and EasyGA framework.
Evaluation: Input/output data fields filled out by EasyAI users should be passed to the EasyNN and EasyGA frameworks so that they may perform operations using said data.

3.1.2.4
Description: The EasyAI web application shall handle input/output forms that the user will fill out or have pre-filled in values.
Evaluation: Input/output forms should pass data filled out by the user to the EasyNN and EasyGA frameworks.

3.1.2.5
Description: The EasyAI web application shall build a virtual environment in the background to allow users to input code without corrupting the site.
Evaluation: The created virtual environment should run in the background without corrupting information on the actual site.

3.1.2.6
Description: The EasyAI web application shall track and interact the user's virtual environments folder with the database.
Evaluation: The user's virtual environments folder should be saved in the database and be visible in the backend.

3.1.2.7
Description: The EasyAI web application shall have a user profile for the user.
Evaluation: There should be a database accessible by the admin page that shows user profile information for each user.

3.1.2.8
Description: The EasyAI web application shall have an admin section that handles all users and permissions.
Evaluation: The admin page should be able to create, modify, and/or delete users, as well as modify their permissions.

3.2 Interface Requirements

- < Describe the interactions of the system with other entities. Interface requirements include a precise description of the protocol for each interface:
    - what data items are input
    - what data items are output
    - what are the data type, the format, and the possible range of values for each data item? (i.e. what is the "domain" of this data item?)
    - how accurate must each data item be?
    - how often will each data item be received or sent?
    - timing issues (synchronous/asynchronous)>
    - how many will be received or sent in a particular period?
    - how accurate must the data be?

- ...>

3.2.1 EasyNN

3.2.1.1
Description: The Neural Network shall only accept float tensors as input.
Evaluation: Check if the Neural Network casts inputs to float tensors to ensure correct types and shapes.

3.2.1.2
Description: The Neural Network shall return a float tensor as output.
Evaluation: Check software for operations used and if they result in tensors.

3.2.1.2.1
Description: The non-Convolutional Neural Networks shall return a float vector as output.
Evaluation: Check software for operations used and if they result in vectors.

3.2.1.2.2
Description: The Convolutional Neural Networks shall return a 3D tensor as output.
Evaluation: Check software for operations used and if they result in 3D tensors.

3.2.1.3
Description: The Optimizer shall set their hyperparameters through initialization.
Evaluation: The Optimizer initializers should accept hyperparameters as parameters.

3.2.1.4
Description: The Optimizer shall take the current iteration, values, and derivatives as input for updating the values.
Evaluation: Check software for Optimizer and Machine Learning usage.

3.2.1.6
Description: The Neural Network shall accept a topology for initialization based on the type of neural network.
Evaluation: Check Neural Network software and documentation on how to define its topology.

3.2.2 EasyAI

3.2.2.1
Description: The EasyAI system shall run multiple users asynchronously.
Evaluation: The EasyAI system should handle multiple users using the site at one time.

3.2.2.2
Description: The EasyAI system shall handle the input data types by not allowing incorrect data types at the form input layer of the website.
Evaluation: Check to verify the data types entered are valid data types and do not pass them to the EasyNN/GA framework if not

3.2.2.3
Description: The EasyAI system shall have an interface that is easy to understand how to use the system, while also allowing the user to understand how the inputs affect the all overall system.
Evaluation: The site should be straightforward and also contain instructions and information on how the site works.

3.3 Physical Environment Requirements

- < Describe the environment in which the system must operate. Physical environment requirements include:
    - type of equipment/environment on which the system must run
    - location of the equipment
    - environmental considerations: temperature, humidity, ...
    - ...>

- Someone fill in the physical server information.

EasyAI

- EasyAI will have a server component that will run the Django website once a stable version is running but for the most part, all the development and advancement will be run on localhost on the developer computer.

3.4 User and Human Factors Requirements

- <Describe the users and their constraints:
    - What different types of users must the system support?
    - What is the skill level of each type of user? What type of training and documentation must be provided for each user?
    - Do any users require special accommodations (large font size, ...)
    - Must the system detect and prevent misuse? If so, what types of potential misuse must the system detect and prevent?
    - ...>

- The EasyAI system shall support any skill leveled python programmer.

- The EasyNN system shall save the model's data to the database.

3.5 Documentation Requirements

- <Describe what documentation is required:
    - on-line, printed, or both?
    - what is the assumed skill level of the audience of each component of documentation?
    - ...>

- The code shall follow PEP8's formatting standards.
- Classes, attributes, methods, and functions shall be documented using doc-strings and type-hints.
- The systems shall use duck typing instead of enforcing type hints.
- The built-in help() function shall print documentation for classes and functions.
- The GitHub repository shall include documentation in the wiki.
- In-line comments shall be used where determined necessary during code review.
- Source code documentation shall assume the reader is proficient with python.
- GitHub documentation shall assume the reader is proficient with python.
- Programmers not on the EasyNN development team should be able to construct a model with the following topologies by following the documentation

3.6 Data Requirements

- <Describe any data calculations: what formula will be used? to what degree of precision must the calculations be made? >
- <Describe any retained data requirements: exactly what must be retained?
- ...>

3.6.1 EasyNN

3.6.1.1
Description: The Machine Learning system shall store the current iteration, and Optimizer, and a tensor for values and derivatives.

3.6.1.2
Description: The Neural Network shall store the Machine Learning system.

3.6.1.3
Description: The Neural Network shall store a collection of Neurons.

3.6.1.4
Description: The Neurons shall store weights, biases, and their derivatives as references/pointers to values in the Machine Learning system.

3.6.1.5
Description: The Neural Network shall use its respective feedforward and backpropagation formulas as found in the literature.

3.6.1.6
Description: The EasyNN system shall use NumPy's linear algebra operations to evaluate large matrix/vector calculations e.g. norm, dot product, and matrix multiplication.

3.7 Resource Requirements

- <Describe the system resources:
  - skilled personnel required to build, use, and maintain the system?
  - physical space, power, heating, air conditioning, ...?
  - schedule?
  - funding?
  - hardware/software/tools?
  - ...>

3.8 Security Requirements

- <Describe any security requirements:
  - must access to the system or information be controlled?
  - must one user's data be isolated from others?
  - how will user programs be isolated from other programs and the operating system?
  - how often will the system be backed up?
  - must the backup copies be stored at a different location?
  - should precautions be taken against fire, water damage, theft, ...?
  - what are the recovery requirements?
  - ...>

- EasyAI shall handle user information. Django has a lot of pre-built handling of user creation frameworks to handle security so that is what we will rely on.

3.9 Quality Assurance Requirements

- <Describe quality attributes:
  - What are the requirements for reliability, availability, maintainability, security, portability ...?
  - How must these quality attributes be demonstrated?
  - Must the system detect and isolate faults? If so, what types of faults?
  - Is there a prescribed mean time between failures?
  - Is there a prescribed time the system must be available?
  - Is there a maximum time allowed for restarting the system after a failure?
  - What are the requirements for resource usage and response times?
  - ...>

- Easy AI will have to do error checking for input to prevent incorrect user interaction with the web application.

- EasyAI We will also have to test the frameworks using unit testing to see how the system handles large data tasks.

**Section 4: Supporting Material**

**4.1.** List of desired models to be supported:

- 4.1.1. Non-dense feed-forward
- 4.1.2. Dense feed-forward
- 4.1.3. Non-dense recurrent
- 4.1.4. Dense recurrent
- 4.1.5. Markov chain
- 4.1.6. Convolutional
- 4.1.7. Neuroevolving non-dense feed-forward

**4.2.** List of optimization algorithms to be supported:

- 4.2.1. Gradient descent
- 4.2.2. Momentum-based gradient descent
- 4.2.3. Stochastic gradient descent
- 4.2.4. Perturbed gradient descent
- 4.2.5. Adagrad
- 4.2.6. Adadelta
- 4.2.7. RMSprop
- 4.2.8. BFGS
- 4.2.9. Broyden's method
- 4.2.10. PDF
- 4.2.11. SR1
- 4.2.12. L-BFGS