Implementation Phase Task Distribution

Ibrahim Bouriga

January 9, 2019

Contents

1 Introduction

Please note: The distribution is made basically on the estimated time of work and not on the number of classes. Many of the classes are abstract and contains only abstract methods. If you find the following distribution not convenient or even unfair, please let me know before the first meeting on Friday. If you have any preferences we can discuss that also.
Fell free to add as much classes as you need, the **deviation** from the class diagramm - which it will be the result of new added classes and interfaces - will be described in the implementation document anyway, but try to avoid that if it is not really necessary.
Each Class has its owner, write your names as author in the documentation if you participated in writing it, so the supervisor knows who to address on the day of the presentation.

1.1 Working with branches

By developping we will face always problems joining our works. That's why we should work on **different branches** (e.g. Avoid pushing directly on master) to avoid confrontation with conflicts when merging branches.
Please choose meaningful names for your branches espicially ***don't name them after your name*** (e.g. Ibrahim's Branch XD).
If someone has a problem with his code (e.g. compiler error, bugs, ...) try to tackle it, if not succeeded just **open an issue on github**, so others could help finding a solution. See the following Link to know how to open an issue in Github https://help.github.com/articles/opening-an-issue-from-code/
Do not forget to **push every major changes** you made in order to keep track of your progress and not to lose data. Avoid pushing too much when it's not needed (e.g. adding a line of comment)

We have to pay attention on those Details not only to make communication between us easier but also to provide the supervisors a clear picture of our work

2 Distribution

View, see section **3**: Bahaa
FPGA, see section **4**: Andres, Ibrahim
Image Model and Neural Network Topology, see section **5** and **6**: Mehyar
Layers, see section **7**: Linjuan, Mehyar

Activation Functions, see section 8: (*Not assigned yet*)

3 View

We refer here to the classes and interfaces that represent the graphical user interface including controller part which handles the view

3.1 Classes and Interfaces in Class Diagram

- HepicsGUI
- WelcomeWindow
- MainWindow
- ControlSection
- PlattformModeLabel
- ImageLabel
- ObserverManager
- ResultObserver
- StateObserver

4 FPGA

4.1 Scheduler

This part includes:

- Scheduler
- Classifier
- Resource Manager

4.2 Mode

All three modes:

- High Performance
- Low Power
- Energy Efficency

4.3 Workers and Work Units

4.3.1 Workers

- FPGA worker
- CPU Worker
- File Load Worker

4.3.2 Work Units

- Layer Work Unit

- Image Load Work Unit

5 Image Model

This part includes all classes related to image management:

- Data Server
- Image Manager
- Image

6 Neural Network Topology

- Neural Network
- Classification Assistant
- Topology
- Topology Displayer

7 Layers

- Input Layer
- Convolutional Layer
- Max Pool Layer
- Response Layer
- Dense Layer

8 Activation Functions

- ReLu
- Softmax
- Tanh
- Sigmoid