Configuration Manual

MSc Research Project
Data Analytics

Abhishek Minz
Student ID: x21173184

School of Computing
National College of Ireland

Supervisor: Vladimir Milosavljevic

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Abhishek Minz
Student ID:	x21173184
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Vladimir Milosavljevic
Submission Due Date:	14/08/2023
Project Title:	Configuration Manual
Word Count:	688
Page Count:	3

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Abhishek Minz
Date:	18th September 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Abhishek Minz
x21173184

1 Development environment

The implementation of the project was done using **Python 3.8.10** and **Keras v2.11.0** with **TensorFlow v2.13.0** as the backend. Training of the models was done on a cloud GPU the specifications for the same are:

Table 1: Hardware Specifications

GPU	1x NVIDIA A10
VRAM per GPU	24GB
vCPUs	30
RAM	200GiB

2 Packages and Modules

Lambda Cloud[1] IDE (Jupyter Notebook) was used for model building and training, therefore '.ipynb' python notebook files are provided. To run the code using Jupyter Notebook, Google Colab, or other similar notebook platforms the **ipynb** module would have to be installed using "!pip install ipynb" in order to import '.ipynb' files like regular python modules using the syntax "from ipynb.fs.full.'filename' import 'module name'", which isn't necessary when running '.py' files using regular IDEs where it can be simply done using "from 'filename' import 'module name'". Some of the other modules and packages required are **albumentations** for data augmentation, **tqdm** for progress bar, **Numpy** for scientific computing, **numxpr version 2.7.3 or higher** for fast numerical expression evaluation, **scikitimage** and **open-cv** for image processing, **scikit-learn** for train test split, **matplotlib** and **seaborn** for data visualization. Apart from this, some standard packages and modules were used from the Python library such as the **os** to interact with the operating system, **random** for pseudorandom number generation, and **glob** for pathname pattern expansion.

3 Files and folders

Because of the length of the source code, it has been divided into several different Python files for easier debugging, DataPreprocessing.ipynb, DataTransformation.ipynb,

[1] `https://cloud.lambdalabs.com/`

Train.ipynb, DataLoader.ipynb, unet.ipynb, custom_unetv20.ipynb, ablationv1.ipynb, ablationv2.ipynb and ablationv3.ipynb.

There are a total of 3 folders, the first folder **stage1_train** contains the raw data (images and corresponding masks)[2]. The other two folders **data** and **new_data** are generated after execution of **DataPreprocessing.ipynb** and **DataTransformation.ipynb** respectively. The files and folders need to be in the same directory for proper execution.

4 Source Code

4.1 U-Net

Replication of the classical U-Net architecture as proposed by Ronneberger et al. (2015) to accommodate 256x256 pixel images.[3]

4.2 Proposed Model

The channel attention TensorFlow implementation used in the proposed model was taken from the GitHub repository[4] [5] and the implementation of the res path has been taken from the official GitHub implementation[6] of Multiresunet Ibtehaz and Rahman (2020).

4.3 Loss Function and Evaluation Metric

The source code for the dice loss and Jaccard index(intersection over union) has been taken from the official implementation[7] of UNet++ Zhou et al. (2020).

5 Program Execution

For program execution, the files have to be run in a particular sequence. First the **DataPreprocessing.ipynb** file has to be run for data pre-processing, the **DataTransformation.ipynb** has to be run for data transformation and lastly, the **Train.ipynb** has to be run for model training

5.1 DataPreprocessing.ipynb

The code is used to pre-process the raw images and masks present in the **stage1_train** folder. The operations performed were merging the masks, and resizing the images and masks. The output of the code is generated in a folder named **data** with the subfolders **images** and **masks** each containing 670 files.

[2] `https://www.kaggle.com/competitions/data-science-bowl-2018/overview`
[3] `https://paperswithcode.com/method/u-net`
[4] `https://github.com/kobiso/CBAM-keras/blob/master/models/attention_module.py`
[5] `https://paperswithcode.com/paper/cbam-convolutional-block-attention-module`
[6] `https://github.com/nibtehaz/MultiResUNet`
[7] `https://github.com/MrGiovanni/UNetPlusPlus/blob/master/keras/helper_functions.py`

5.2 DataTransformation.ipynb

The output data generated by **DataPreprocessing.ipynb** is used to generate augmented images and their corresponding masks and are saved in the folder **new_data** with subfolders **images** and **masks** with 3,350 files each.

5.3 Train.ipynb

This contains the primary code of the project and is used to make function calls to the following files

1. **DataLoader.ipynb**: This module is used to fetch the augmented files in the folder **new_data** generated by **DataTransformation.ipynb**.
2. **unet.ipynb**: Implementation of vanilla U-Net
3. **custom_unetv20.ipynb**: Implementation of the proposed model
4. **ablationv1.ipynb** Implementation of U-Net with channel attention
5. **ablationv2.ipynb** Implementation of U-Net with channel attention and MDR Block
6. **ablationv3.ipynb** Implementation of U-Net with channel attention and Res Path

References

Ibtehaz, N. and Rahman, M. S. (2020). Multiresunet : Rethinking the u-net architecture for multimodal biomedical image segmentation, *Neural Networks* **121**: 74–87.
URL: *https://www.sciencedirect.com/science/article/pii/S0893608019302503*

Ronneberger, O., Fischer, P. and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation, *in* N. Navab, J. Hornegger, W. M. Wells and A. F. Frangi (eds), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer International Publishing, Cham, pp. 234–241.

Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N. and Liang, J. (2020). Unet++: Redesigning skip connections to exploit multiscale features in image segmentation.