Software Requirements Specification

for

Face-mask Detection in Real-time

Version 1.0 approved

Prepared by Ahmed Mostafa Kamel,
Eslam Taha Abd-Elrhim
Hossam Ahmed Thapit

Assiut University – Faculty of Computers and Information

Apr 4, 2021

Table of Contents

Table of Contents ii
Revision History ii
1. Introduction 1
1.1 Purpose 1
1.2 Document Conventions 1
1.3 Intended Audience and Reading Suggestions 1
1.4 Product Scope 1
1.5 References 1
2. Overall Description 1
2.1 Product Perspective 2
2.2 Product Functions 2
2.3 User Classes and Characteristics 2
2.4 Operating Environment 3
2.5 Design and Implementation Constraints 3
2.6 User Documentation 3
2.7 Assumptions and Dependencies 3
3. External Interface Requirements 4
3.1 User Interfaces 4
3.2 Hardware Interfaces 4
3.3 Software Interfaces 4
3.4 Communications Interfaces 4
4. System Features 5
4.1 System Feature 1 5
4.2 System Feature 2 (and so on) 5
5. Other Nonfunctional Requirements 6
5.1 Performance Requirements 6
5.2 Safety Requirements 6
5.3 Security Requirements 6
5.4 Software Quality Attributes 6
5.5 Business Rules 6

Revision History

Name	Date	Reason for Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of Face-mask detection in Real-time software. It will explain the purpose and features of the software, the interfaces of the software, what the software will do, and the constraints under which it must operate.

This document is intended for **institution administrators** and **potential developers**.

1.2 Document Conventions

This document was created based on the IEEE template for System Requirement Specification (SRS) Documents.

1.3 Intended Audience and Reading Suggestions

- Institutions administrators such as in malls, hospitals, universities …etc.
- Developers who are interested in working on the project by further developing it or fix existing bugs.

1.4 Product Scope

Face-mask Detection in Real-time is a software that helps users to automatically identify whether someone or a group of people are wearing a face mask or not with the type of that mask in a real-time fashion.

It can be used mainly in large institutions, especially governmental institutions for analytical purposes.

Some institutions could use it for medical purposes such as preventing the current pandemic of "covid-19" from being worse by limiting the people's abilities to do something without a mask such as buying something in the stores, entering the university campus ...etc.

This is a software for **Custom Object Detection**, a paradigm that appeared in the **Computer Vision Applications** field of research relating to **Deep Learning**.

1.5 Reference

- IEEE Std 1233, 1998 Edition, IEEE Guide for Developing System Requirements Specifications.

- IEEE Template for System Requirement Specification Documents: https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc

2. Overall Description

2.1 Product Perspective

Face-mask Detection in Real-time is a software developed to be used in/be part of institutions for monitoring and analytical purposes.

The users of the system can use it for controlling the entrance into the institution, calculate some analytical measures or even use it to allow/prevent someone with/from some privileges inside the institution.

It is a product built using Python, an open-source high-level and general-purpose programming language, and PyTorch, an open-source machine learning library based on the Torch library.

2.2 Product Functions

Train:

- --data_dir: Define a directory which contains the data.
  - Default = 'Face Mask Dataset'
- --save_dir: Define a save directory for checkpoints as a string.
  - Default = 'saved_models/checkpoint.pth'
- --arch: Pick a Pre-trained Model Architecture.
  - Default = densenet121
- --learning_rate: Choose the learning rate of your optimizer.
  - Default = 0.003
- --hidden_units: Choose the hidden units of your first hidden layer.
  - Default = 512
- --epochs: Choose the number of epochs.
  - Default = 15
- --gpu: Train your network using a gpu.

Predict:

- --model_path: Specify a file that contains the model state.
  - Default = 'saved_models/checkpoint.pth'
- --image_path: Specify an Image file to be used in the prediction process.
  - Default: 'Face Mask Dataset/Test/WithMask/1175.png'
- --top_k: Specify the number of top K likely classes to be displayed.
  - Default = 2
- --gpu: Inference using a gpu.

2.3 User Classes and Characteristics

- **Administrators:** for monitoring and controlling.
- **Security Guards:** for managing institution entrance.
- **Data scientists:** for researches and data analytics purposes.
- **Developers**: for working on the project by further developing it or fix existing bugs.

2.4 Operating Environment

- Win 7, Win 8, Win 8.1, Win 10.

2.5 Design and Implementation Constraints

- Environment:

  Face-mask Detection in Real-time is developed using Python and the wide variety of libraries that python offers, such as, Numpy, torch, cv2 …etc.

  It is inevitable that you will have to run this system in an environment that does support python.

- Hardware limitations (computing power):

  Another constrain is the computing power. The training process has been done on Kaggle, a data science community that offers access to GPUs on its cloud.

  Minimum requirements for training would be something like:

Parameter	**GPU**	**GPU Memory**	**GPU Memory Clock**	**Performance**	**No. CPU Cores**	**Available RAM**	**Disk Space**
	Nvidia P100	16GB	1.32GHz	9.3 TFLOPS	2	12GB	5GB

- Regulatory policies:

  The data used in this project is published under the Apache 2.0 License.

2.6 User Documentation

To be developed later.

2.7 Assumptions and Dependencies

The ability to provide a GPU in case of retraining the model, otherwise the only dependency is the hardware that the model will be deployed on it, such as security cameras, mobile devices … etc.

3. External Interface Requirements

3.1 User Interfaces

To be developed later.

3.2 Hardware Interfaces

The minimum hardware requirements of Face-mask Detection in Real-time software:

- For Inference only in real-time:

Parameter	CPU	CPU Freq.	No. CPU Cores	CPU Family	Available RAM	Disk Space
	Intel(R) Xeon(R)	2.3GHz	2	Haswell	12-16	25GB

- For both Training and Inference:

Parameter	GPU	GPU Memory	GPU Memory Clock	Performance	No. CPU Cores	Available RAM	Disk Space
	Nvidia P100	16GB	1.32GHz	9.3 TFLOPS	2	12GB	5GB

These requirements are for the best performance with no latency issues, less or more computing power is applicable depending on the datasets being used, training objective, and inference process.

3.3 Software Interfaces

Face-mask Detection in Real-time software requires Python 3.7 and Torch 1.8.1 and CUDA 10.2 for GPU usage.

Further information to be found on: https://pytorch.org/

3.4 Communications Interfaces

Face-mask Detection in Real-time software requires an internet connection to install the libraries used to build it, update already installed ones.

Internet connection is only required once, otherwise, the software can work without internet.

4. System Features

4.1 Train

4.1.1 Description and Priority

The ability to train a model with a specified pre-trained architecture along with a dataset of your choice to be classified.

The user can also specify the learning rate of an optimizer, specify the hidden units in the first layer, and whether to train on a CPU or GPU.

This function priority is medium as it's a critical feature.

4.1.2 Stimulus/Response Sequences

```
$ python train.py --data_dir ../Face Mask Dataset --save_dir saved_models/checkpoint.pth
--arch densenet121 --learning_rate 0.003 --hidden_units 512 --epochs 15 --gpu
```

4.1.3 Functional Requirements

Review Section 3.2

4.2 Predict

4.2.1 Description and Priority

The ability to Inference the model with a specified data item.

The user can also specify whether to Inference the model using a CPU or GPU.

This function priority is high as it's the core of the software, using a trained model to immediately start detection is the core of the software.

4.2.2 Stimulus/Response Sequences

```
$ python predict.py --model_path saved_models/checkpoint.pth --image_path
../Face Mask Dataset/Test/WithMask/1175.png --top_k 2 --gpu
```

4.2.3 Functional Requirements

Review Section 3.2

5. Other Nonfunctional Requirements

5.1 Performance Requirements

This system is working with data in a real-time fashion, it should be designed very efficiently to handle any circumstances where the data feed into the software is very large.

5.2 Safety Requirements

As it considered to be a monitoring software, the only safety required is to save the model parameters to be used later. The software provides both functionalities to save/load a model.

5.3 Security Requirements

There are no security requirements as the software doesn't save any data by itself. It might be required if the application is integrated later with custom analytics software.

5.4 Software Quality Attributes

The Correctness of the software depends on the quality of the data and the network architecture. The software is packed with a ~99.1% Accuracy. Training a custom model on different data is the responsibility of the user if he decided to.

The software is flexible and maintainable as it provides a range of options to train and Inference a model.