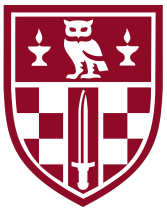
Birkbeck, University of London

Pneumonia Detection from Chest X-Ray Images

Author:
Baran Buluttekin

Supervisor:
Dr. George Magoulas



A project report submitted in fulfillment of the requirements for the degree of MSc Data Science

in the

Department of Computer Science

June 21, 2020

Declaration

I have read and understood the sections of plagiarism in the College Policy on assessment offences and confirm that the work is my own, with the work of others clearly acknowledged. I give my permission to submit my report to the plagiarism testing database that the College is using and test it using plagiarism detection software, search engines or meta- searching software.

The report may be freely copied and distributed provided the source is explicitly acknowledged.

Abstract

New test text. New words

Contents

1 Introduction

Medical diagnosis and specifically computer aided diagnosis (CAD) is a hot topic in the field of technology. One of the main reasons of becoming a hot topic is the recent innovation and breakthroughs achieved by computer vision research. Combined with poor healthcare coverage around the globe, CAD systems offer a promising solution to mitigate devastating impact of the fatal diseases such as pneumonia. Controversial topics such as whether or not artificial intelligence will replace the radiologies in the future aside, these automated systems can offer answers for patients questions in absence of medical help or to very least offer much needed second opinion in the face of unsatisfied diagnoses. Given all mentioned possible benefits of the CAD systems, this project is focused on building classification CAD system for diagnosing pneumonia from the chest X-ray images.

1.1 Aims and Objectives

Aim of this project is to build a fully functional chest X-ray image classification pipeline that implements CI/CD principals to experimentation and deployment.

1.1.1 Objectives

Project will be implemented with execution of fallowing objectives:

- **Carrying out general data exploration:** This part involves general check on dataset.
- **Data pre-processing and augmentation:** Preparing the data for model ready state.
- **Building baseline model with well known neural network architectures:** This step involves setting additional benchmarks with out of the box models from section 4.
- **Using pre-trained network to increase model performance:** Using pre-trained networks to help training and accuracy of the model.
- **Visualizing neural network to ensure learning quality:** For making sure model learning as intended and focusing on correct parts of the image.

- **Model ensembling:** Using ensemble method with different neural network architectures.

- **Applying different deployment options:** Implementation of different deployment options. Based on their trade offs.

1.2 CI/CD Pipeline

In this section, I will give a brief introduction CI/CD pipeline to explain what CI/CD is and why it is chosen as a preferred way of build this project.

Continuous integration (CI) is a workflow strategy that helps ensure everyone's changes will integrate with the current version of the project in the typical software engineering team. This lets member of the team to catch bugs, reduce merge conflicts, and increase overall confidence of your software is working. While the details may vary depending on the development environment, most CI systems feature the same basic tools and processes. In most scenarios, a team will practice CI in conjunction with automated testing using a dedicated server or CI service. Whenever a developer adds new work to a branch, the server will automatically build and test the code to determine whether it works and can be integrated with the code on the main development branch. The CI server will produce output containing the results of the build and an indication of whether or not the branch passes all the requirements for integration into the main development branch. By exposing build and test information for every commit on every branch, CI paves the way for what's known as continuous delivery, or CD, as well as a related process called continuous deployment. Difference between continuous delivery and continuous deployment is that CD is the practice of developing software in such a way that you could release it at any time. When coupled with CI, continuous delivery lets you develop features with modular code in more manageable increments. Continuous development is an extension of continuous delivery. It's a process that allows you to actually deploy newly developed features into production with confidence, and experience little, if any, downtime. Even though benefits of using CI/CD pipelines more prominent in the software teams, integration automated testing will help even individual project such as this by reducing time for debugging.

In more granular detail, this system works with central version control services and in this project central version control service used is Github. GitHub uses communication tool called *webhooks* to send messages to external systems about activities and events that occurred in the project. For each event type, subscribers will receive message related to the event. Generally events refer to action involving the software such as new commit push or pull (merge) request and more. In this case whenever new commit is pushed to the any branch of the project, message from Github will be send out to third party system called *travis* [1] and travis will fetch the most recent version of the project and run the tests associated with it. When test runs completed with the latest version of the software, test results

[1] https://travis-ci.org/

will send back to relevant commit as status information. Travis is a hosted CI service that allow build and test software hosted in version control services. Upon completing the build and test travis then passes the test results via Github API which will be visible to the developer.

1.3 Engineering Related Challenges

Role of hardware accelerators. Debugging related challenges of Neural Networks

1.4 Project layout and design

Overall project design such as folders and library interactions.

1.5 Reproducibility Guidance

Reproducibility is a fundamental

As a scientific project, it is very important that anyone can reproduce the experiments and findings in this project to verify the conclusions reached are accurate. [1]

to reproduce the experiments and findings in this project to verify the conclusions reached are accurate.

Parts to consider before reproducing. Repo is not public Most runs carried out in colab notebooks should be positioned in same level as the src library running shell utility files requires kaggle api key

2 Data

Reminder of the dataset. [2]

2.1 Data Augmentation

Data augmentations applied and example results of the augmentation.

2.2 Limitations of the Dataset

Issues related to validation set size.

2.3 Data Procession

Information about tf.data processing pipeline and advantages compare to other data feeds.

Talk about the decisions for image size and how it relates to information loss and preservation. Briefly mention the high variance of the image sizes and image resizing options and choice. Discussing different representation of the dataset will be covered.

References

[1] Daniel Kermany and Kang Zhang. *"Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification"*. 2018. DOI: http://dx.doi.org/10.17632/rscbjbr9sj.2.

[2] *Open Access Biomedical Image Search Engine*. https://openi.nlm.nih.gov/. Accessed: 2019-03-12.