Anomaly detection for treatment planning and a learning health system in radiotherapy

Computer Integrated Surgery II

Spring, 2018

Daniel Yuan and Vincent Qi

Mentors: Dr. Todd McNutt, Pranav Lakshminarayanan

Introduction

- Our primary goal was to improve the integrity of clinical radiation therapy data. By improving the quality of clinical data available to physicians, we can minimize the risk involved with radiotherapy for cancer patients.
- We developed a fully commented API and a code framework that can help identify potentially anomalous data with statistical detection and analysis. The framework allows for the modular insertion of various detection rules.
- We also developed some of our own detection modules based off of previous research. These allowed us to explore areas where anomalies occur. They also serve as examples for future users.

The Problem

- A large proportion of patient radiotherapy data is collected manually by physicians. For example, contour model slices are individually collected and spliced together. Daily patient assessments are filled out and then manually inputted in a database.
- Data collection is a repetitive and tiring process, making it prone to human error. Example include missing contour slices and inputting incorrect patient variables.
- Data is also being collected on a massive scale. The amount of data collected means that going through a database manually is a difficult task.
- While error detection methods exist they often have limited use due to either being too specific or being not flexible enough for use on other databases that method was not built for.

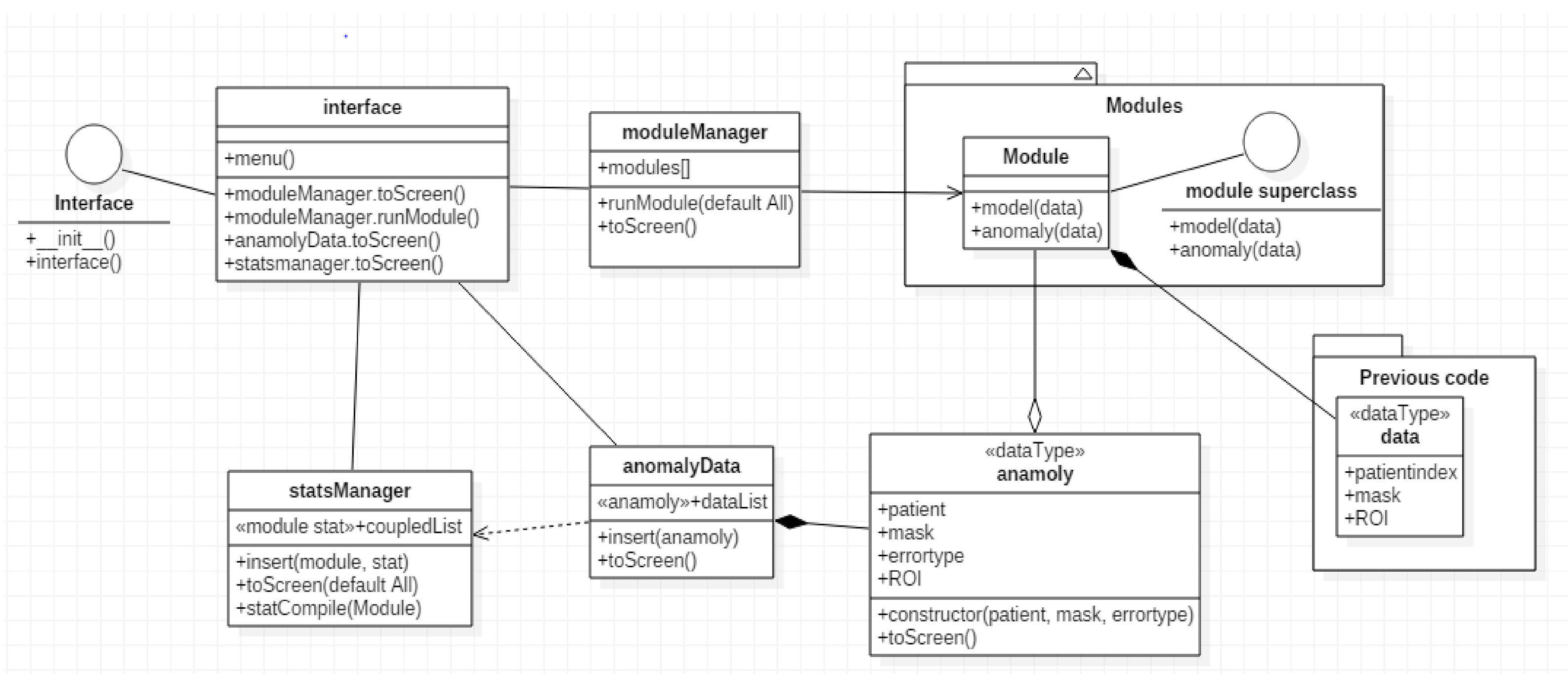

Figure 1: UML diagram of code framework

Our Solution

- We designed a framework that allows for the modular insertion of various detection rules in order to allow active an approach using constantly updated clinical databases.
- We tested several validation algorithms, such as continuity and dosing maps maximums.
- Our API is fully documented so that future users can easily write their own integrity checks and run it over the database.

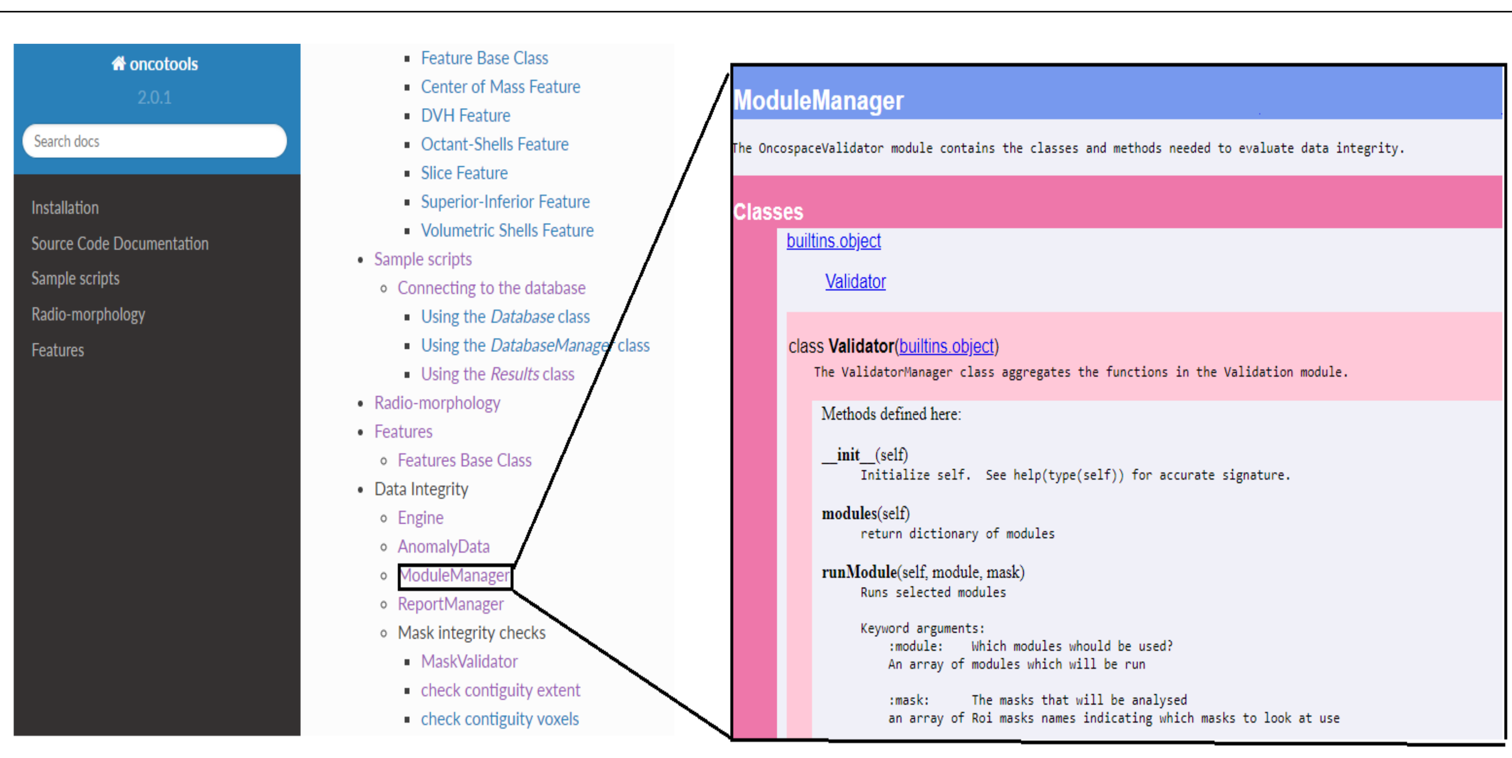

Figure 2: Example of documentation

Code Implementation

- We implement the framework structure in Python. Python is used because it allows for simple portability and OS compatibility. It also has many existing packages that can be used to augment and improve the framework.
- The framework is based off of oncotools, a prebuilt library by the McNutt lab for querying the oncospace database.

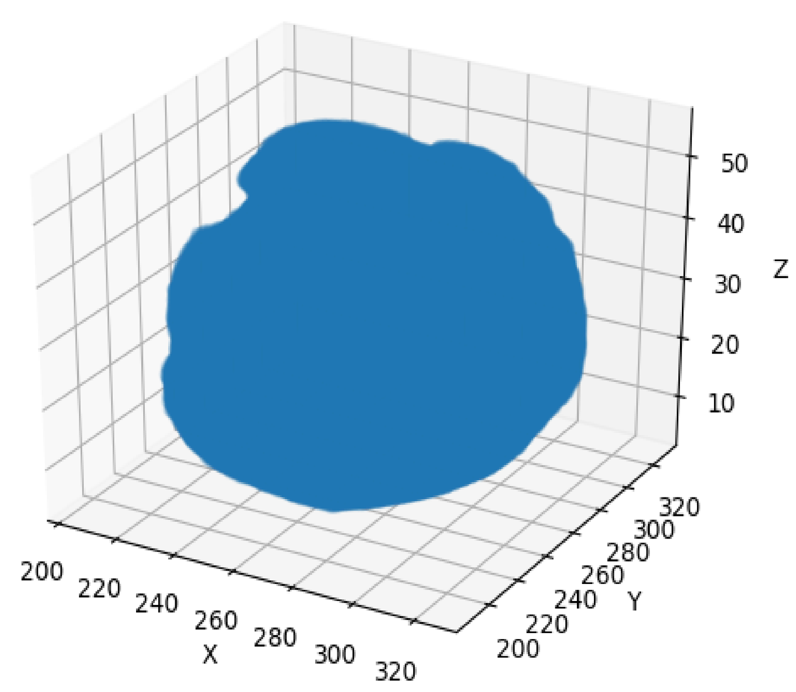

Figure 3: Example of Mask

Patient	Data with no error	Errors found
0	35	2
1	40	1
2	25	2
3	39	3
5	26	1
6	40	1
7	41	6
8	16	0
9	40	1

Figure 4: Example of table of errors found for individual patients

Outcomes and Results

- We were able to create a framework that allows for the insertion of various detection algorithms. Most of the process is automated. The user only needs to write the actual detection method with a structured input and output format.
- We were able test our framework with several of our own detection algorithms in order to validate that it works.
- We were able to query the database for over a thousand patients and determine the existence of anomalous contour maps. We also were able to determine which patients had anomalous dosing maps and questionable assessments.

Future Work

- Since the API works, the next major step would be to implement more error detection modules and enable physicians to analyze more types of erroneous data.
- Increase flexibility of framework to enable output beyond error/no-error, such as a scale on the likelihood of a data point to have errors.
- Optimize the code for efficiency for running over a large database. For example, implement parallelism and improve query structures.

Lessons Learned

- The more flexible a code needs to be, the harder it is to implement. To many complicated relationships in the framework can cause confusions and errors.
- Traversing across a database is highly memory intensive and a balance of memory usage and speed in required for optimization.

Credits

- Programing and design: Vincent Qi
- Programing and design: Daniel Yuan

Support and Acknowledgements

- Thanks to Todd Mcnutt and Pranav Lakshminarayanan for being our mentors
- Thank you to Dr. Russ Taylor for leading the class