**Your project title: Which can be longer when it is displayed on the front page of the document**

|  |  |
| --- | --- |
| Report Name | Outline Project Specification |
| Author (DAF16) | Daniel Luke Fryer |
| Supervisor | Supervisor Name (abc) |
|  |  |
| Module | CS39440 |
| Degree Scheme | GH76 (Artificial Intelligence and Robotics) |
|  |  |
| Date | February 5th 2016 |
| Revision | 0.1 |
| Status | Draft |

# Project description

The aim of the … project is to research, implement and improve on pre-existing algorithms which aim to rank genes within the genome. Genes are ranked by importance within the genome for specific experiments, for instance, if there is a new potential cure for a disease then it is helpful to know which genes are being affected, what that effect means and how strong the effect is.

Genes can be ranked based on expression data from microarray experiments, whereby genes are, up-regulated, down-regulated or there is no change. Up-regulation means that there has been a fold-increase in expression, e.g. a two fold increase means that the gene doubles production, of for instance rNA. Genes which express more are considered to be of more importance. If a drug changes the expression of a gene then it is to be noted by a higher ranking than unchanged genes.

Ranking based solely on expression data does not however build up the perfect picture of importance within the overall network. A gene which does not change its expression levels but does activate its transcription factor will affect genes which are turned on by this transcription factor. This gene then, affected by the drug could be responsible for turning on many other genes which change expression. Based solely on expression data only the genes which are turned on and up or down-regulate will be ranked highly. Whereas clearly the gene responsible for turning them on should also have a high ranking.

In order to account for such genes within the ranking system it is important to include the network structure data for the genome. This is achieved by using Gene Ontology (GO) data, one network is created for each section of the GO Biological Process, Cellular Component and Molecular Function, whereby genes are connected if they share an annotation defined by these networks [1]. The underlying network of genes is combined with expression data from the given experiment to create a ranking system less susceptible to noise which will include the above mentioned gene types in the ranking. As noted within the paper, there are other sources of gene network data and measures which can be used instead of, or on top of GO data.

The algorithm will be implemented in Python using open sourced elements of the original algorithm. Once implemented and tested the algorithm will be experimented with and extended. It is possible to look at alternative approaches and incorporate these into the algorithm as well as the possible application of such an algorithm to other, similar areas of biology.

# Proposed tasks

The following tasks will be performed on this project:

Research into the original algorithm along with alternative algorithms. This will be expanded onto research into expansion of the algorithm along with alternative uses of the algorithm.

Documentation and github

Implementation and then expansion of the original algorithm which combines the use of expression data with the GO network data to create a sophisticated ranking system.

Project meating + journal

Mid term presentation + final demonstration

# Project deliverables

# Initial annotated bibliography

# Bibliography

|  |  |
| --- | --- |
| [1] | R. B. D. J. H. a. D. R. G. Julie L Morrison, “GeneRank: Using search engine technology for the analysis of microarray experiments,” *BMC Bioinformatics,* 2005. |

**This paper fully works through how the GeneRank algorithm works, with the algorithm, explanations of expression data and GO data as well a synthetic networks used to prove their algorithm. Shows proof of concept, with it being used on a yeast expreiment and improving previous ranking techniques.**