**Enhanced network GeneRank**

**Abstract**

Microarray experimentation is done to determine the expression change between a reference and a “treated” sample where the reference will be based on a normal state and the treated could be from genetic changes or drug treatments. These experiments give a lot of statistical data based on the results. This can be combined with prior knowledge of the genes biological function to give more useful information about the results. GeneRank along with other ranking algorithms attempt to automate and improve on this process by creating a prioritised ranking list of genes based on the expression change in the experiment as well as network information of the genes.

GeneRank successfully improves upon ranking solely on expression change by integrating network data into the algorithm. The type of network information, as well as the amount of information directly affect how much of an improvement is made. Although clear improvements on ranking can be shown GeneRank is far from the best or most efficient algorithm for this and has been surpassed by modern machine learning algorithms. GeneRank suffers most on large scale experiments containing thousands of genes due to the structure of the algorithm.

For its time GeneRank was a clever and innovative idea which clearly benefited biological research by producing better ranking measures. However, as time has gone on others have taken the idea and improved upon it while avoiding some of the short falls that GeneRank suffers from.

**Introduction**

**Human Genome Project (HGP)**

The HGP was first publically promoted by Renato Dulbecco in an article published in 1984 who argued that sequencing the human genome would greatly enhance the understanding of cancer [1]. This historic goal was achieved in 2001 by a collaborated effort of 20 different groups. It was at the time, the largest genome to be extensively sequenced [2]. This achievement paved the way for a new era of biology with continuous efforts to sequence new genomes. The effects can be seen clearly in medicine where new approaches are being developed which rely on knowledge and understanding of genetic variation [3]. In the fourteen years since the sequencing of the human genome, vast amounts of data covering various genomes has been collected and given rise to large, open-access databases [4]. A direct result of this is the increasing use of Bioinformatics as a tool to manage and exploit this new found biological information. Bioinformatics expanded due to the human genome project, however, while there has been clear progress, bioinformatics is yet to be fully utilised [5].

**Model Organisms**

The term model organism is historically given to species such as the mouse, because they are small and have short generation timeframes [6]. Such organisms are usually fairly simple and allow for simulations of biological processes found in other organisms, in the case of medical science these processes are thought to be shared by humans [7]. The genomes of these organisms have been sequenced, meaning that there is vast amounts of knowledge about the genome and biological processes within it. The mouse organism has a database containing such information [8]. The mouse is widely used as a model organism as it has the common characteristics of a model organism, and as a mammal, has stronger similarities to the human genome. The mouse genome has roughly 22,000 genes, a number close to the roughly 20,000-25,000 genes in humans. The exact number of genes in the human genome is unknown and widely debated [9].

Furthermore the mouse is commonly used as a model organism as it is more ethical to conduct research on mice than it is to do the same research on species closer to humans, such as monkeys. The combination of the above characteristics of mice is the reason that the mouse is widely used as a model organism for medical and biological science.

Yeast is another model organism, while not as closely related to humans as mice, yeast still shares many genes with humans [10]. David Botstein *et al*. discovered that for 31% of the potential protein encoding genes within the yeast genome there is “a statistically robust homolog among the mammalian protein sequences” [10]. The yeast genome has been sequenced, and has roughly 6000 genes. Yeast has a very rapid reproductive cycle and with a smaller genome has been used widely for cell biology research [11]. The term model organism has evolved to include organisms which share unique traits or characteristics which are deemed worthy of research [6].

The two other commonly used model organisms are the common fruit fly *D. melanogaster* and the roundworm *C. elegans* [6] [12] [13]. These model organisms share the common traits of model organisms and were amongst the first organisms to have had their genomes sequenced [13].

**Microarray Experiments**

Microarray experiments allow for profiling of gene expression and can in medical science give vital information of the specific features of a disease [14]. Today, expression level data is commonly used to explore causes and treatments for cancer. For this purpose, as well as for all other uses of expression data there is a dire need to process the vast quantities of data [14] [15].

A large use of microarray experiments, is to find the fold-change of gene expression between two experiments with one being a reference state and one the test or treated state. The data which comes out however, needs to be processed. There is a real risk that due to the scale of data coming from microarray experiments that based on bad statistical analysis wrong conclusions could be drawn from the data [16].

In many experiments, expression data is required from large numbers of tests in order to give validity to results. Over the course of several different treatments the tests performed can reach into the hundreds. The resulting expression data contains a lot of noise, as for instance, the same experiment performed on the same tissue sample could give different expression level results at different times. Different expression levels on the same tissue sample has also been observed when using different microarray technologies [16]. The preparation of tissue samples and the use of different dyes can also affect results.

As a result of noise, the significance of fold-changes in expression levels is also not always clear [16]. The change in expression values can however coincide with observed characteristics of a disease as found by S Hanash and C Creighton [14]. Although expression changes discovered by microarray experiments can lead to valuable information backed up by statistical significance, combining the expression results with prior knowledge of network data or observed characteristics leads to more founded results.

Therefore, results from microarray experiments may be sufficient alone in certain areas however, for many functions prior knowledge is needed as well as expression results are not perfect. This is due to the noise and uncertainty that is inevitable in results from microarray experiments. The use of prior knowledge increases confidence that the results are being interpreted properly with statistical and biological significance being accounted for.

The issues from large scale microarray experiments and the data produced is well known and there are various methods used to make sense of and interpret large scale data. For instance there are methods used to deal with missing values as well as various bioinformatics approaches which manipulate large scale data to reveal patterns and biological processes [17] [18]. There are also various databases containing expression data for organisms, such as the mouse [19]. Bioinformatics approaches to extract the valuable information that comes from microarray experiments are continually being developed and improved.

**PageRank**

The success of the Google search engine was largely determined by the success of PageRank [20], the algorithm used to determine the order in which webpages are displayed after a search. This order is based on a combination of key words used within the search as well as connections between websites and web pages. In this instance connection between pages was defined by hyperlinks. For every web page there is a node in the graph with an out degree and in degree defining how many pages the page links to and how many pages link to that page [21]. If purely the terms searched for were used, the volume of results would be staggering. PageRank is used to prioritise the results of the given search. The more hyperlinks linking from other web pages to a web page containing the key search terms, the higher the ranking given.

**GeneRank**

The purpose of GeneRank as well as all subsequent ranking algorithms is to prioritise genes for further analysis based on biological experiments [22]. This is one of many ways of getting valuable information about biological process from the large scale expression data which comes from microarray experiments. An example use would be to determine a prioritised list of genes within the human genome, which have an expression change between a reference and a treated experiment. This could give vital information on the processes which occur when a treatment is used, for example, to treat cancer. With the example of cancer, there is a high number of candidate genes which are thought to alter cellular processes [23]. The purpose of algorithms such as GeneRank is to give a faster and better prioritisation of such candidate genes which require more in depth research. This is achieved through automation of the process.

The idea was put forward by Morrison *et al.* to adapt the approach used for PageRank, in order to rank genes based on the connections of genes within the genome[22]. GeneRank combines the expression level of the genes with a connectivity matrix. When ranking is based purely on expression data for a given experiment the ranking will not give the full picture. Some genes will be neither up-regulated nor down-regulated however their transcription factors could be activated. As a result some genes will be activated as they are controlled by these transcription factors. These genes, which affect the up-regulation and down-regulation of other genes, are clearly important and in some respects should be ranked more highly than those which only change expression level. These genes are accounted for by GeneRank but not by ranking based only on expression levels[22]. This is why the combination of prior knowledge in the form of connection data with expression data is such a powerful concept which improves gene ranking and continues to be used and developed by subsequent algorithms.

The connectivity matrices are built from connectivity data which can come from a range of sources. For the algorithm, If two genes are connected then the associated element within the matrix is set to 1, if the two genes are not connected then this is set to 0. The algorithm has a weighting value d, which determines the influence connection data has on the final ranking list. If d is set to 0 then the algorithm bases the ranking purely on expression data, If d is set to 1 then the ranking is based purely on connection data, and if d = 0 these two elements are given equal weighting.

It is important to note that for the GeneRank algorithm to work effectively the absolute expression value must be used, therefore, if the expression change is negative as a result of down-regulation it must be converted to a positive number. As a result an expression value of -1 would be changed to +1. This allows the algorithm to rank based on total change as without this addition down-regulated genes would always be ranked lower than up-regulated genes.

**Limitations of GeneRank**

GeneRank had limitations in its approach in using this connection data. The main approach suggested and implemented with testing and evaluation was to use GO (Gene Ontology) annotations as a source of connection data. This approach has been used previously by others with success [14] [24]. Two genes are connected if they both share the same GO annotation. As mentioned by Eugene Demidenko [25], this can become an unrealistic approach to use in practice as large numbers of genes require massive connectivity matrices. As an example, with the yeast genome, containing 6,000 genes, 6million connections can be found when using GO annotation. This leads to a graph of around 286KB in size, unreadable by programs designed to show large scale graphs visually. The creation of such a graph takes several minutes as every genes GO annotations is matched against ever other genes GO annotations.

The scaling problems for larger genomes continue into the algorithm itself, which updates the ranking of every gene n times, with n being the number of genes. For the first iteration all genes are updated against the initial ranking of the first gene. For the second iteration all genes are updated against the ranking of the second gene. This means that for iteration n all genes are updated according to the rank of the next gene at iteration n-1. This means that the algorithm itself can take several minutes when 6,000 genes are used. An additional side-effect of this approach is that the order in which the genes are entered into the algorithm will affect the resulting prioritised ranking list. This idea is further shown in demonstration below where the mouse genome containing 22,000 genes was used. GeneRank is better suited to smaller genomes, as the larger the genome the longer the algorithm takes exponentially and the worse the resulting ranking down to order of genes.

The difficulty in using GeneRank for each new experiment is also discussed by Eugene Demidenko, as for each experiment there is new expression data and so the algorithm has to run again, which could take a long time over multiple experiments [25].

Morrison *et al.* suggest that when d is set to 0.5 it produces the best results and so should be used as the standard weighting for the algorithm[22]. Eugene Demidenko also however, states that having the threshold parameter d set to 0.5 is unjustified [25]. Further evaluation within this study shows that in fact a value of 0.95 or 1 for the weighting d gives the best results on different data. This is similar to results that Morrison *et al* discovered using synthetic data where d values of 0.75 to 0.85 performed best, they also note that d = 0.85 is reportedly used by Google for PageRank[22]. It is curious then that for general use they determine that a d value of 0.5 is suited best, stating that it improved ranking while no deterioration was observed. A d value of 0.5 is however, provably non-optimal and perhaps not best suited for most uses of the algorithm.

**Evaluation and proof within GeneRank**

There are two evaluation methods used within the GeneRank paper, a synthetic network and Correlation Coefficient Networks as shown in the data set [26]. Using the synthetic networks they prove that ranking with a high weighting on connection data is significantly better than ranking based solely on expression levels. This is done by comparing two sets of genes, where it is known that one set, setA should be ranked higher than the other, setB. When compared during ranking via the GeneRank algorithm a receiver operating characteristic (ROC) score result is used [27] [28]. The ROC value given is 0.5 if the two sets are mixed and 1 if perfectly separated. Therefore in this experiment the higher the score, the better the ranking as setA contains genes known to be higher than those in the setB.

The “Correlation Coefficient Networks” evaluation is used to support the idea that GeneRank has given a better ranking measure when connectivity data is included and highly weighted. This in itself is an achievement however it does not suggest that genes within each set, for instance in setA which is known to be ranked higher than genes in setB, are ranked well. It could be that although genes in setA are generally ranked higher than those in setB as they should, genes in setA may not have a good ranking within the set.

It is clear that GeneRank can run, and perform well on genomes of up to 6,000 genes such as the yeast genome. The algorithm used provably gives a better prioritised ranking list of genes than the common approach at the time of using only expression values for the same ranking. This was perhaps one of the first algorithms to achieve this end and lead to continued improvements and alternative ranking algorithms [25][29]. A MatLab implementation of the algorithm remains free and open access allowing others to use it for research or improve upon it. Since the success of GeneRank alternatives with use similar approaches out perform this, however, this is to be expected as time passes. GeneRank remains a good option for small scale genomes and is easy to implement and use.

**Enhanced Network GeneRank**

The benefits of having a better prioritised ranking list of genes from experiments as discussed above are of huge value to biological and medical research. As such there are continuous efforts to improve on the idea of GeneRank. The purpose of Enhanced network Generank is to evaluate the GeneRank algorithm against modern approaches and to implement and suggest further improvements. This is done by following the basis of experimentation done within the GeneRank paper and combining that with modern evaluation techniques with proven alternative approaches to ranking genes [29].

The GeneRank algorithm is re-written in Python using Anaconda 3.5 [30] as the original is written in MatLab which requires a user license. The algorithm is embedded in a program designed for ease of use. This is done so that the algorithm is more accessible, and comes with several different versions. The aim is to provide a user friendly library containing the GeneRank algorithm which will produce a prioritised ranking list of genes as well as a graph based network file which is viewable using Cytoscape [31]. This allows for visualisation of the highly prioritised genes within the network structure of the genome which has been identified and used for the ranking.

**Modern Evaluation Techniques**

With the current evaluation methods used, it cannot be determined whether individual gene rankings can be considered good, but more so that the overall ranking of all genes is improved upon. Therefore instead of using the same evaluation methods, Enhanced Network GeneRank focuses on the use of the methods used by Daniela Nitsch *et al.* [29]*.* This modern approach is used to evaluate several machine learning approaches to the gene rank problem in order to determine which one produces the best rankings. This allows the GeneRank algorithm to be directly compared to other approaches in order to determine whether or not GeneRank produces rankings good enough to warrant its use in the future.

For this evaluation 40 subsets of genes within the mouse genome are used. The mouse is a model organism as described above, and is used in biology frequently including use as a model for development of drugs developed for human use [6]. As a result, the mouse genome is a good test case for GeneRank and all alternative ranking measures. If it is possible to effectively rank genes within the mouse genome for a given experiment or treatment then this could lead to more effective and targeted research. The mouse genome consists of around 22,000 genes, for the evaluation of various ranking methods, 40 subsets of 100 genes were used[29]. This favours algorithms such as GeneRank which do not perform so well on large scale genomes however remains a fair evaluation as this type of experiment with 100 to 1,000 genes are target at one time due to prior knowledge of the genome.

Each subset contains expression data for 100 genes based on an experiment where one gene was knocked out, called a KO Gene. Various biological research projects use KO mice [32] [33]. This is an area in which ranking algorithms could aid in biological research. The 100 genes used for each evaluation are the “nearest neighbours” to the KO gene[29]. The expression values for all genes within the subset is therefore based on this KO gene. As a result the ranking of the KO gene should ideally be within the top 10, as the KO gene caused the other genes changes in expression levels.

The number of times the KO gene is in the top 10 ranked genes, over all 40 experiments gives an indication of the algorithms performance. As a second measure the same count is done for the number within the top 20 ranked genes and as a third measure a ROC score is taken for each subset. The ROC score is calculated with every KO genes expected ranking score set to 1 with all others set to 0 for the GeneRank algorithm, all ranked genes should have a score between 0 and 1, the higher the score, the higher the ranking. The ROC score compares the expected outcome against the real ranking score produced for each gene in the subset. For further analysis of the GeneRank algorithm, these three evaluation scores, top 10 count, top 20 count and roc scores, were measured across all values of d. Where 0 <= d <= 1.0 in increments of 0.05.

The average ranking across the 40 files is another indication of how the algorithm performs. The methods used by Daniela Nitsch *et al.* largely achieved an average ranking of below 10 with the best algorithm having an average rankinf of 8[29]*.* In comparison the best average ranking the GeneRank algorithm produced, across all values of d and several different networks, was 25. This is however, much better than the average ranking of 42 which is produced when only expression data is used as a ranking measure.

**Observations from Evaluation**

Experimentation on using different network data within GeneRank shows that although different networks have some effect on ranking, the difference on average ranking is minimal. However, the network used can have a larger impact on whether the gene is ranked in the top 10 genes or not. Overall within GeneRank it is clear that for ever increasing values of d, the weighting for network information, the better the average ranking and likelihood of a gene being in the top 10 increases dramatically. This continues as a trend until d is 0.95 or 1, with some networks giving better scores for a d value of 0.95 while others give better rankings when d is 1 and so the expression data isn’t considered at all. When networks are combined, GeneRank gives a better ranking. Overall, there is clear evidence that GeneRank does improve upon the system of ranking purely on expression data. This shows, that although not as good as modern approaches, GeneRank performs the function it was designed to do, and does so consistently across different networks.

**Connection data**

The connection data for a genome can come from a variety of sources, the main focus of the GeneRank paper is on GO annotations [22]. Noting that GO annotations have been successfully used in previous papers [14] [24]. Other methods such as Protein-protein interactions have been also used previously [34] [35]. GO annotation ID’s for genomes can be found on the GO consortium website [36]. From here files containing the genes along with their GO ID’s can be found for many different organisms, including many model organisms. Connection by GO annotations, if two genes within a genome share a GO term then the two genes are connected. On a genome such a Yeast with 6000+ genes this gives around 6 million connections which as discussed above, can be unrealistic to create and use in real practice due to size.

The use of GO Ontology data is a more proven concept and is used for analysis of the algorithm within the Synthetic data section of the GeneRank paper [22]. GO Ontology is split into three sections defined by the GO Ontology Consortium [36]. The three Ontologies are: Cellular Component, Biological Process and Molecular function. Each Ontology has GO ID’s associated, genes which share the same GO ID’s for example Biological Process, will share the Biological Process defined by that GO ID. For this network structure genes sharing a GO ID should be connected within the network. This allows for three unique networks to be created as well as the possibility of combined networks where network information from all three, or any combination of the three can be used. Again, downloads for these networks can be found on the GO Consortium website.

The use of multiple networks based on different connection data allows for more in-depth testing of the algorithm and could lead to better results, if for instance one network is proven to be better than others. During the analysis of the 40 subsets of data used in this study, when all three ontologies were combined the graph size ranged from 24-63KB which is a manageable size.

Another source of connection data is Protein-Protein interactions whereby two genes are connected by protein interactions [37]. The data showing the connections by protein-protein interactions comes from the String database [38]. The connection can be based on a number of sources and measures, such as experiments, databases, neighbourhood, textmining, co-expression, co-occurance and gene fusion. Each connection found by these sources and measures will have an associated confidence score. The higher this score is, the more certain it is that a connection between the two genes exists.

Protein-Protein interaction based networks can be used as standalone networks or combined with the above GO Ontology networks. There are also other methods for connection genes based on prior knowledge, however, these were not used for evaluation of the GeneRank algorithm and usually require in-depth study or the use of databanks [38] [39]. The purpose of all gene network and connection based projects is to further understand the interactions between genes and the biological processes involved. For this end, the more connection data available for a given genome, the more prior knowledge available. It is this prior knowledge which modern ranking algorithms use to create a better prioritised ranking list for a given experiment.

**Degree of Certainty**

It is important to note that for both gene expression data and gene connectivity data there is an associated degree of certainty. As discussed above, noise within gene expression data can reduce certainty about results. This however, remains true for connection data, the connections found between genes, using any of the methods defined above will have a degree of certainty. Some connections between genes are more certain than others. For this study the certainty of connections is not ensured, and so may not be fully reliable. Due to the size of the networks that are used as well as the combination of network data and expression data, this should not affect the end prioritised ranking list in a meaningful way. This is however an area which could be expanded on and evaluated in itself. Where networks with higher levels of certainty could potentially change the end rankings.

**Alternatives Approaches**

Several subsequent studies have shown ways to improve upon GeneRank itself, as well as using the same principle idea of combining expression data with connection data with other machine learning approaches [25][29]. There are different ways in which it is possible to incorporate both data types in order to produce a prioritised ranking list. The algorithms are also affected by the quality of connection data used as well as the expression data. Gene Ranking and especially network ranking as a method has been used by various other projects [40] [41]. Many of these approaches focuses on reducing the false discovery of connections, for this reason Eugene Demidenko states that “gene connectivity can be adequately expressed in terms of the gene pairwise squared correlation matrix” [25] [42] [43]. Microarray data is usually correlated to a phenotype, however as connectivity can come from other sources this is not always required. Phenotypes can be examined after the ranking results.

It is clear that since the implementation of this style of ranking introduced by Morrison *et al.* further study and improvements have proven how useful it is for biological research [22]. There are however, other methods that achieve the same end goal, of focusing research on prioritised elements within a genome. This can be achieved by the study of networks, sub-networks and clustering. By looking at the network structure and clusters within that structure, it is possible to identify important genes as well as important sub-networks [44]. Network clusters can be created for a given experiment or for the reference state of the genome. Both can give great insight into the biological process [39].

**Methods**

**Expression Data**

The expression data came from 40 publically available datasets from Affymetrix chips on mice with and without KO genes as used by Daniela Nitsch *et al.* [29]***.* (pre-processing of data and what logFC is)** Table 1 shows the data sets used for evaluation.

Table 1

**Network Creation**

The network creation is somewhat dependant on the structure of the network data. For all GO data, for both GO annotations and the three GO networks the same method is used. Each gene has an element in a list containing all its associated GO ID’s. This list is in the form of [GeneName, GO ID’s, GeneName, Go ID’s,…] and contains all genes and their GO ID’s in the order they appear in the file read. Using this structure, each gene has its GO ID’s compared to that of every over gene, if one of the GO ID’s matches then the genes are connected and a link is added in the graph.

The protein-protein interaction network was created using the String 10.0 dataset [38]. This was done using the “Multiple Proteins” section, with the ~100 genes from each of the 40 KO gene files put in the list of names section. The Mus Musculus organism was selected as these files all contain genes from the mouse organism. Continuing on the table produced was exported. For the use of protein-protein interaction networks within this study, only experiments was used as a source for network connections. This however, could be at any confidence value.

For the protein-protein network, the data file which is read in has two columns, each column has a list of genes. For each line the genes in column 1 and column 2 are connected in some way, it is possible therefore to add a link in the graph for these two genes very simply. Al network connections were created using the add\_node function found in the networkx package [45].

An example network graph is shown below in figure1. This is the network graph for the file for the KO gene Abca1 using the protein-protein interactions data.

Figure 1.

**GeneRank Algorithm**

The main aspect of the algorithm is a python implementation of the original GeneRank algorithm designed by Morrison *et al.* which is defined below. This ensures that the evaluation of the algorithm is based on the same algorithms put forward. The main aspect of change is the data put in, the network construction, evaluation technique and supporting code which allows for easy repeatable use of the algorithm. The algorithm is based on the following:

**https://static-content.springer.com/image/art%3A10.1186%2F1471-2105-6-233/MediaObjects/12859_2005_Article_558_Eque_HTML.gif** [22]

As suggested by Morrison *et al.* all genes are given an initial ranking **r** [0] = **ex**/||**ex**||1, where ||·||1 denotes the vector 1-norm. Ex is the absolute value of the expression change for the gene. Wij is based on connectivity whereby, if the genes i and j are connected then a value of 1 is given and if they are not connected a value of 0 is given[22]. In place of the connectivity matrix used Morrison *et al.* the connection data is stored in a graph created using the Networkx tool [45].

**Python Implementation of GeneRank**

The python implementation of GeneRank was developed using the list data structure which allows for the method to be used as a standalone method or as part of the script created.

geneRank(geneNameList, exprDataList, normExprDataList, G, d) where G is the graph created using networkx and d is the weighting value. The three lists are must be in the same gene order, where element [i] for each list refers to information for the same gene.

sumOfConnectionList = [] // two lists for the other required values rankingValueList = [] num3 = len(geneNameList) // all the list have the same number of ellements

i = 0 j = 0

while (i < num3): // two while loops create rj[n] while (k < num3): if (i == 0): // create the other two required lists on the first run through sumOfConnectionList.append(0) rankingValueList.append(normExprDataList[j]) // this gives each gene their initial ranking score elif (i != j): // as long as i and j aren’t the same gene if there is an edge between genes i and j in the graph set hasEdge to 1 else set hasEdge to 0 if gene i has no connections then temp\_connection = (hasEdge \* (rankingValueList[i-1])) / 1 else temp\_connection = (hasEdge \* (rankingValueList[i-1])) / (G.degree(geneIDList[i]))

sumOfConnectionList[j] = sumOfConnectionList[j] + temp\_connection connectionValue = (1-d)\*exprDataList[j] rankValue = connectionValue + d\*(sumOfConnectionList[j]) rankingValueList[j] = rankValue

j = j + 1

i = i + 1 j = 0 // reset j so that every gene is compared again next iteration

return rankingValueList // The final prioritised ranking list

https://static-content.springer.com/image/art%3A10.1186%2F1471-2105-6-233/MediaObjects/12859_2005_Article_558_Equf_HTML.gif

With sumOfConnectionList being the equivalent of for each gene.

Some genes within the KO gene files had multiple expression level results. In this case the average expression value was used.

**Evaluation**

The evaluation is made up of an ROC analysis as well as the average rank and top10 counts for each value of d over the KO gene files. For the ROC analysis each KO gene is given an expected score of 1 and all others are given an expected score of 0. This is build up in a list containing expected scores for all KO gene files. Another list of the true rankings is also created, these two lists are used for the roc analysis after being converted to being np arrays. The roc\_auc\_score is defined in sklran.matrics [46]. The average rankings and top10 counts are simply calculated based on the output data file created, this is shown in more depth in design and implementation.

**Flow Chart**

Figure 2 and figure 3 Show flow chart representations of the full process, from reading the data files to getting the end rankings. For figure 1 the process is described for a real use scenario. Figure 3 shows the process for evaluation techniques.

Figure 2.

Figure 3.

**Experimental Design**

**Original Design**

The original design relied on the same datasets used by Morrison *et al.* In order to get an idea of whether or not the algorithm functioned according to the existing study [22]. This was going to be used as a basis for design as well as an evaluation of the methods used by Morrison *et al.* The gene data for the yeast genome was found (where) and contained the GO annotation information for all genes within the yeast genome(ref for dataset). This was used as a basis to build the python implementation of the algorithm, as the list data structure used had to be constructed correctly for the data being read for later parts of the algorithm. This allowed for a full design of the reading of data, building and adding connection data to the network graph, the geneRank algorithm implementation and the subsequently writing the output to a file.

As part of this design several other sections were required, such as creating the required lists for the geneRank algorithm. As well as the re-ordering of the list containing the genes and their ranking scores so that the first gene would be the highest ranked gene. The initial design did not contain a ROC analysis or average ranking scores. Figure 4 shows the function structure of the original design.

Figure 4.

This design however had to change as it proved difficult to get the real expression data from the files provided by Morrison *et al.* All implementation provided was written in MatLab and although a MatLab to python converter, scipy.io.loadmat was used so that it was possible to access the data structures, it proved difficult to get out the raw expression change data required [47].

**Adapted Design**

The updated design was based on the method shown by Daniela Nitsch *et al.* [29]*.* The creation of the 40 KO gene files containing expression data and GO network information is described in the methods section of this paper. 40 publically available datasets from Affymetrix chips on mice with and without KO genes were used.

This new design had to make use of these new KO gene files which contained the logFC value as well as GO network ID’s for each of the three GO networks. The programme structure designed for ease of use of the algorithm stayed much the same. However, in order to allow for the algorithm to be called for multiple files, and not just one example case some changes were made to the creation and structure of the lists used. The GeneRank algorithm was modified such that it took in multiple lists. The lists are required to be in the same gene order, so each lists elements contain information for the genes in the order they appear in the file which is read in. This required the creation of new lists and such the design changed.

The modification of the original GeneRank method design allowed for it to be called multiple times within the programme. This was required for evaluation across all 40 KO gene files, as was the ROC score. Therefore the design also needed to include a way to calculate the ROC score over the 40 files, for all values of d. All relevant data was to be appended to a file which would end up containing the ranking score for all values of d, across all 40 KO gene files. In order to speed up the process, one design allowed for file names to be entered to read, and write to. However, another design had the KO gene file names and the desired output file name hardcoded in. This made debugging far quicker when the design was implemented. Figure 5 and figure 6 show the amended function design structure for these two variations.

Figure 5.

Figure 6.

In order to get some of the evaluation methods another script was designed to take the output data and calculate the average scores along with the top 10 and top 20 counts. Another file was used to calculate the roc score across all 40 KO gene files, using much of the same design for getting the raw data results. Figure 7 and figure 8 show the design for these scripts.

Figure 7.

Figure 8.

**Evaluation Method Design**

Based on this new design, it was possible to use the same evaluation methods used by Daniela Nitsch *et al.* [29]*.* They use four different evaluation measures to evaluate the various machine learning algorithms implemented. These are the average rank, over the 40 KO gene files, the top10 count and the top20 count along with a roc score. By using the same data files which are used in this study it is possible to directly compare GeneRank to the modern methods described. The new data, along with expected results also means that it is possible to test the running of the algorithm itself, and to evaluate whether or not the algorithm is equivalent to ranking on expression value when d = 0. Over the 40 KO files any duplicate genes had their logFC values averaged, this average was used as the expression change value for the algorithm.

The design based on the 40 KO gene files also allowed for a protein-protein interactions network to be implemented as data coming from (where?) allowed for the creation and use of files showing the various connections by protein-protein interactions. The addition of this network means that a more thorough investigation into GeneRank was possible. It is possible from this to evaluate across 5 different network structures for all values of d across the 40 KO gene files leading to a more robust evaluation than that used by Morrison *et al* [22]*.*

**Scientific Use Design**

**Desired Goal**

The end goal of GeneRank is to allow biologists access to a programme which will automate and thus speed up the process of priority ranking genes. The end result is proven to work however, especially without access to MatLab is difficult to use in practice. Therefore in this study a user friendly design was introduced to allow for easy use of the programme which could read and write files or take in raw data.

**User Friendly Design**

The original design, much like the original evaluation design used the original GeneRank paper as a guidance. Firgure 9 shows the original design which includes the reading of files, using a GeneRank method and outputting the rankings in order. Also included is the creation of the graph file based on GO network information, graphs created will be written to an xml file.

Figure 9.

As the evaluation design changed in order to use the 40 KO gene files, so too did the user design. The new design made use of the GeneRank method which could be called as a standalone, or as part of the programme. This simplified much of the design however as with the evaluation design it meant that when reading files several different lists had to be created. Figure 10 shows the final design which biologists would be able to use. It is written in Python such that anyone with programming knowledge can see the source code and adjust it as necessary. For instance, they may not want to write the xml graph file every time as this can take some time depending on size and may not be necessary in all cases.

Figure 10.

This design focuses on GO network data however it would be simple to change this for other network types. As GO network information is arguably good enough and is widely available it is what this design uses. The design not only gives a full prioritised ranking list of the genes supplied but also writes the xml graph file for the network graph used. This could aid biologists by giving them access to the network structure used as well as the raw rankings. By combining this information it would be possible to see clusters of highly ranked genes in the network. The end goal is to provide as much useful data as possible to allow biologists to have the best idea of where they should focus future research.

**Implementation Implementation**

All implementation was done using the improved design based on the 40 KO gene files. The creation of the networks required as well as the gene expression data are described within the methods section. The end result was several different python files, one designed for ease of use for biologists which prints only the prioritised ranking list. One file to calculate the ranking of each KO gene for each value of d from 0 to 1 in intervals of 0.05. This was based on the GO ID networks. A different file was created to deal with the protein-protein interactions network as this required reading in two files. However, it did not need as many separate features as the connections within these files were simpler to parse. These two files printed, one after another, lines containing the gene name followed by each rank position at each value of d.

A further file took the output of the files containing the rank position of the KO gene, for all 40 files, for all values of d. This then calculated the average rank of the 40 KO genes for all values of d as well as creating a top10 and top 20 count for each value of d. A final file was used to calculate the ROC score for each value of d over the 40 KO gene files.

The implementation was written fully in python using Anaconda 3.5 [30].

**Results**

**The goal**

The aim of this study is to create a user friendly, fully accessible programme which provides the same service as GeneRank. While also, evaluating the success of GeneRank as well as the claims found within the results and conclusions by Morrison *et al.* [22]*.* The Success of GeneRank is then to be measured against modern techniques which use modern machine learning approaches. The results of these approaches come some six years after GeneRank and therefore are more advanced, however, they still use the underlining structure of GeneRank. In that they use the combination of expression data and connection data.

In order to fully evaluate GeneRank against these new approached, the same datasets were used, with the same measures for evaluation shown. Using these datasets and evaluation methods it is also possible to evaluate the claims founded by GeneRank. The claim that the algorithms produced are better at ranking than when only expression data is used, and that it is better than a random ranking.

**General GeneRank**

Shown in Figure 11 is an example output for the GeneRank algorithm which can be used with ease, by biologists. This is the standard output, it is impossible to tell just by this however, how good the ranking list performs. It is unknown as to what the best prioritised ranking list for this example gene set for this experiment is. This is shown only to illustrate what the output from the algorithm looks like in regular use. The proof and evaluation of GeneRank and how good any prioritised ranking list produced will be, follows below.

Figure 11.

**Validation of GeneRank**

In order to evaluate GeneRank as an algorithm in how well it performs in itself and against modern approaches the methods described previously have been used. The largest indication of how well the algorithm performs is the average rank of the KO gene across all 40 KO gene files, along with the top10 and top20 counts. This is, as described previously, calculated for all values of d and done repeatedly over 5 different networks. This gives a good picture of how the weighting value d influences results as well as how the network information used can also affect results. Table 2 shows the overall results for GeneRank across these 5 different networks.

Table 2

**Discussion**

**Critical Evaluation**