35th Annual International Conference of the IEEE EMBS
Osaka, Japan, 3 - 7 July, 2013

An Ensemble Rank Learning Approach for Gene Prioritization

Po-Feng Lee, Von-Wun Soo

Abstract— **Several different computational approaches have been developed to solve the gene prioritization problem. We intend to use the ensemble boosting learning techniques to combine variant computational approaches for gene prioritization in order to improve the overall performance. In particular we add a heuristic weighting function to the Rankboost algorithm according to: 1) the absolute ranks generated by the adopted methods for a certain gene, and 2) the ranking relationship between all gene-pairs from each prioritization result. We select 13 known prostate cancer genes in OMIM database as training set and protein coding gene data in HGNC database as test set. We adopt the leave-one-out strategy for the ensemble rank boosting learning. The experimental results show that our ensemble learning approach outperforms the four gene-prioritization methods in ToppGene suite in the ranking results of the 13 known genes in terms of mean average precision, ROC and AUC measures.**

I. Introduction

To find the most promising genes among a large list of candidate genes in account for a certain disease or function has been defined as the gene prioritization problem [17]. Because there are still so many genes for which we don't know their functions and roles in cells. The traditional in-vitro experiments with large samples from both normal and disease tissues usually demand much cost and efforts. But nowadays, many big biological data are becoming more and more accessible via different kinds of databases such GenBank, PDB, OMIM, etc. In the last decade, several different computational approaches have been developed to solve this challenging problem. For example, prioritizing disease candidate genes rely on connecting network-based data [7][8][10][18] and others integrate multiple data sources to prioritize candidate genes [9][21]. Since very gene prioritization method should have its own strength and weakness [6], we should be able to find a way to combine them together to achieve a better performance by assigning a proper weight of each prioritization method [19]. The ensemble learning is an effective principle to solve such kind of problems [1][3][4].

In this paper, we focus on how to use ensemble learning to combine various gene prioritization methods and to achieve a better performance result. We adopt a Rankboost learning algorithm as a solution to gene prioritization problem [5]. It optimizes gene prioritization for a disease using some known disease genes as training set. However, to improve the gene ranking by ensemble learning, a naïve combination of traditional Rankboost learning algorithms requires intensive calculations [5] that cannot have good performance. Therefore, we improve its efficiency by employing an ensemble learning combination strategy on the Rankboost learning method. Instead of ranking all the genes, we also specify a threshold to limit the scope of computation. Only those genes fall within the scope are to be computed. Another problem we encountered is, due to lack of discriminative ranking information for disease genes as ground truth, the rank scores of many genes by using the original prioritization methods turn out to be the same. Hence, we add a weighting function and adjust the parameters of Rankboost algorithm according to 1) the absolute ranks generated by the adopted methods for a certain gene, and 2) the ranking relationship between all gene-pairs from each prioritization result to improve the result [5]. We use the prostate cancer as the training data and use ToppGene suite and ToppNet as the gene prioritization tools [8][16].

The results turned out to be better than merely adding a combinative weighting function. In the experiment, we found that about half of genes are in correlation with the disease in literature of the 13 training genes in each prioritization result. We also obtain better performance in terms of average precision, mean average precision and area under curve (AUC). It can verify the modified algorithm can really improve the performance which we predicted.

II. Methods

A. Rankboost algorithm for a gene prioritization problem

Rankboost proposed by Freund combines a set of weak ranking features h_t [5]. The goal of the algorithm is to minimize the loss function. So Rankboost loss function is an upper bound [5]. To reduce the ranking loss, Rankboost iteratively selects a ranking feature from a pool of candidate features and calculating the combined weights until the loss falls within certain bound [5]. If a weak rank $h_t(x)$ ranges within the interval [0, 1], we can set its weight α_t based on it performance r_t as defined in step 6 in Fig.1.

Algorithm: Rankboost algorithm
Input: initial distribution weight D over
$X \times X = \{(x_1, x_2), (x_1, x_3), \ldots, (x_{n-1}, x_n), (x_{n-1}, x_n)\}$
Number of learning rounds T.
Output: $H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$
Process:

1. $D_1(i) = D$
2. *Setting the threshold* θ
3. For $t = 1 \sim T$
4. Train weak learner using distribution D_t.
5. Get weak ranking $h_t(x_i) = 1 - \frac{Rank\ of\ x_i}{\text{The total number of genes}}$: $X \rightarrow R$.
6. Choose $\alpha_t = \frac{1}{2}\ln(\frac{1+r_t}{1-r_t}) \in R$.

* The research is partially supported by the Bioresources Collection and Research Center of Linko Chang Gung Memorial Hospital and National Tsing-Hua University of Taiwan R.O.C. under the grant number 100N2722E1.

Paul Lee is with Department of Computer Science, National Tsing Hua University, Hsinchu Taiwan; e-mail: peterken0620@gmail.com).

Von-Wun Soo is with Institute of Information Systems and Applications, National Tisng Hua University, Hsinchu Taiwan. (e-mail: soo@cs.nthu.edu.tw).

7. Update $D_{t+1}(x_0,x_1) = \frac{D_t(x_0,x_1)\exp(\alpha_t(h_t(x_1)-h_t(x_0)))}{Z_t}$

Where Z_t is a normalization factor (chosen so that D_{t+1} becomes a probability distribution).

8. Output: $H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$

Figure1. The procedure of Rankboost algorithm

In Fig.1, D_t is set by an ideal rank defined by the users. Because we don't have precise information about the ideal gene prioritization result for a specific biological problem, we use a weighting function instead of an ideal pair-weight $D_t(x_0, x_1)$. $D_t(x_0, x_1)$ means the weight assigned to the gene pair between the genes x_0 and x_1 which would be increased if the example pair does not belong to a correct class. Such a process is repeated T times. Finally it combines the scores and ignores the sign by T weak learners. Then it gets a final rank sorting by the sum of the combined scores. Due to the setting of x is either 0 or 1, the prioritization would result in the same score for many genes that are hardly to tell their ranking orders, so we change h_t into a value in the interval of [0, 1]. We integrate results of prioritization methods instead of data samples. We improve Rankboost with a threshold value for each *known* gene by using the highest rank from all prioritization methods [5]. It cuts the computation beyond the threshold in pairwise computation and thus improve the computational efficiency.

B. *The weighting function*

The basic idea is to use a novel weighting function that reflects different partial contributions from various weak ranking learners instead of a conventional fixed pair-weight for running the boosting algorithm [2][19][20]. In this study, we proposed a Rankboost learning algorithm with a weighting function, dubbed as "Rankboost_W". At first, we set three parameters s_1, s_2 and s_3 as the following：

- s_1, s_2: The absolute ranks generated by the adopted methods for a certain gene. We set 1 ~ 0.1 according to gene rank 1 ~ 10000 for each interval 1000.
- s_3: The consensus score between all gene-pair from each method. If the rank of gene x_0 is higher than gene x_1 or the rank of gene x_1 is higher than gene x_0 in all four tools we set into 1 and if the rank of gene x_0 is higher than gene x_1 or the rank of gene x_1 is higher than gene x_0 in only three tools we set into 0.5. If the rank of gene x_0 higher than gene x_1 or the rank of gene x_0 is higher than gene x_1 in only two tools we set into 0.

Then the simple weighting function is defined in Eq.(1):

$$\sigma(x_0, x_1) = (s_1(x_0) + s_2(x_1) + |s_3|)/3 \quad (1)$$

Then we add it to r_t in Rankboost [5] as in Eq. (2):

$$r_t = \sum_{x0,x1} D_t(x_0, x_1)\ \sigma_t(x_0, x_1)\big(h_t(x_1) - h_t(x_0)\big) \quad (2)$$

Because we only know the genes we left out are disease causing genes but we don't know the ranking order relation between these genes, the algorithm would produce many zeros in $D_t(x_0,x_1)$. But if most $D_t(x_0,x_1)$ are set to zero, the overall weighting function is hard to produce the discriminant effect. So we set $D_t(x_0,x_1)$ are following:

$$D_t(x_0, x_1) = \begin{cases} D_t(x_0, x_1) & s_3 > 0 \\ -D_t(x_0, x_1) & s_3 < 0 \\ 0 & s_3 = 0 \end{cases} \quad (3)$$

We set the pair weight $D_t(x_0,x_1)$ other than zero only when s_3 is not equal to zero. This kind of setting allows the algorithm to be merely affected by the weighting function of the pair. For example, $D_t(x_0,x_1)$ will obtain some weight while the ranking score $D_t(x_0,x_1)$ would not be affected if $s_3 > 0$.

C. Weighted combination of weak prioritization methods

Finally the algorithm would find α_t and h_t that are combined to a final score for each gene by the voting result of the weighted sum of all α_t and h_t. Then we obtain the final ranking result by sorting the scores of all the genes. The overall process with its system architecture is shown in Fig.2 The four prioritization methods adopted are denoted as ToppGene, ToppNet1, ToppNet2 and ToppNet3, respectively.

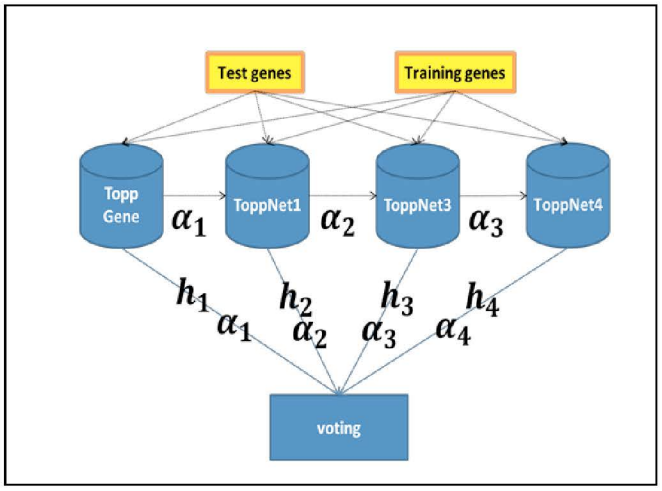

Figure2. System Architecture

III. RESULTS

A. *Selection of training genes, test genes and training genes and test genes*

We choose ToppGene suite as the learning tools [16]. The reason we choose it is because it contains both ToppGene (data and text mining gene prioritization methods) and ToppNet (K step Markov, Hits with Priors, PageRank with Priors: network-based gene prioritization methods) [8]. We combine these tools and use different type of methods.

B. *Selection of training genes and test genes*

The 19 genes selected as training examples are all known classical markers for prostate cancer in OMIM [11]. To avoid redundant data and ignore many genes that we have less information about their functions in NCBI [13], we used human protein-coding genes from HGNC [15].

C. *Left-one-out*

We use left-one-out method to justify whether the disease causing genes can be ranked higher than the original ranking result as shown in Fig.3. The idea for the experiment method is like cross-validation but it's a little different. At first, we should obtain each ranking result by left-one-out method. It

means we move out genes from disease genes to human gene as candidate genes. Then we use the training genes and candidate genes to train the prioritization tool [9]. Then we recorded the ranking result for each method. If we assign equal weight to each prioritization method denoted as Equal_W then we obtain a result as shown in the last column of Table I after combining the ranking results. The rank of each gene falls at the average position of all prioritization methods as we thought. The number in the brackets means the relative rank of the tool compared to the others and we sum the relative ranks of all genes into the SUMMARY field. After running our algorithm, we can get the result of Table II. For example, the Gene Symbol AR, it means we left AR gene from 13 training genes to the prioritization method. The rank is 33 when we run the ToppGene suite [16]. For the other 3 tools, we can get the result each is 27, 15 and 35. The other genes are similarly processed.

Although the results were mixed, more than half of the genes in Rankboost_W outperformed those in ToppGene. And the ranks for all the genes are better than the result for ToppNet (K Step Markov), ToppNet (Hits with Priors) and ToppNet (Hits with Priors). We observed that although many genes which Rankboost_W were not marked as number one performer with respect to other methods, it at least was marked as the second place with respect to the others. So we rank the ranking results for each prioritization method and account the sum of them in the summary field. Clearly, the result showed the Rankboost_W is better than the others.

In our conjecture, the most likely explanation for some genes that did not perform better than the rank generated by Toppgene is because the genes order at the top part in each prioritization method may be inconsistent. So it implies that the weighting function for s_3 did not show its discriminant effect as expected. However, this result yielded limited information about our study.

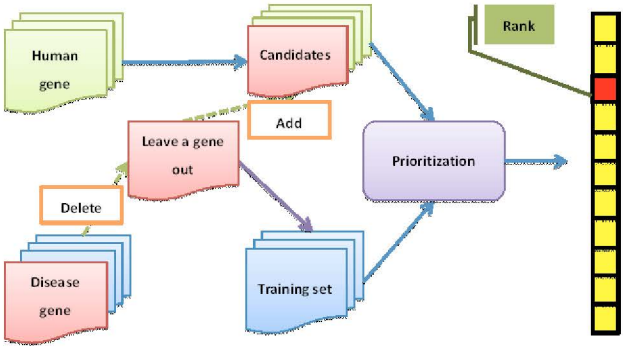

Figure3. LEFT-ONE-OUT EXPERIMENT METHO

TABLE I. THE RESULT FOR LEFT-ONE-OUT (EQUAL_W).

Gene Symbol	The prioritization method				
	ToppGene	*K Step Markov*	*Hits with Priors*	*Page Rank with Priors*	*Equal_W*
AR	33(4)	27(2)	15(1)	35(5)	28(3)
ZFHX3	144(1)	3496(4)	4044(5)	3148(3)	2708(2)
BRCA2	139(2)	165(3)	238(5)	136(1)	170(4)
CDH1	21(1)	402(4)	158(2)	424(5)	252(3)
KLF6	25(1)	1746(4)	585(2)	1859(5)	1054(3)
HIP1	263(1)	2199(3)	2470(5)	2309(4)	1811(2)
CD82	1332(2)	1549(4)	755(1)	1664(5)	1325(3)
MSR1	52(1)	8181(5)	6219(3)	8079(4)	5633(2)
MXI1	42(1)	2272(4)	2439(5)	2015(3)	1692(2)
PTEN	2(1)	871(5)	341(2)	733(4)	487(3)
MAD1L1	480(3)	400(2)	1181(5)	385(1)	612(4)
CHEK2	335(4)	243(2)	532(5)	192(1)	326(3)
ELAC2	834(1)	3768(3)	4039(5)	3890(4)	3133(2)
SUMMARY	23	45	46	45	36

TABLE II. THE RESULT FOR LEFT-ONE-OUT (RANKBOOST_W).

Gene Symbol	The prioritization method				
	ToppGene	*K Step Markov*	*Hits with Priors*	*Page Rank with Priors*	*Rankboost _W*
AR	33(4)	27(3)	15(2)	35(5)	8(1)
ZFHX3	144(2)	3496(4)	4044(5)	3148(3)	74(1)
BRCA2	139(3)	165(4)	238(5)	136(2)	29(1)
CDH1	21(1)	402(4)	158(3)	424(5)	133(2)
KLF6	25(1)	1746(4)	585(3)	1859(5)	280(2)
HIP1	263(1)	2199(3)	2470(5)	2309(4)	680(2)
CD82	1332(3)	1549(4)	755(2)	1664(5)	413(1)
MSR1	52(2)	8181(5)	6219(3)	8079(4)	45(1)
MXI1	42(1)	2272(4)	2439(5)	2015(3)	633(2)
PTEN	2(1)	871(5)	341(3)	733(4)	154(2)
MAD1L1	480(4)	400(3)	1181(5)	385(2)	237(1)
CHEK2	335(4)	243(3)	532(5)	192(2)	89(1)
ELAC2	834(1)	3768(3)	4039(5)	3890(4)	1670(2)
SUMMARY	28	49	51	48	19

D. Performance measure

We used all 13 genes as training genes and integrate the results from each prioritization method. We obtain the gene prioritization results after running each prioritization method. In the ROC curves indicates whether the 1265 prostate cancer genes we can found from PGDB (Human Prostate Gene Database) [12], NCBI (National Center for Biotechnology Information) [13]or KEGG (Kyoto Encyclopedia of Genes and Genomes) [14]. The different between these databases and OMIM is the genes relative to the disease are identification by biologist. According to in-vitro experiments but the other database are not.

We obtain a better result in terms of ROC curves as shown in Fig.4 and AUC, mean average precision (MAP) as shown in Fig.5 respectively. The MAP is 0.2710 and AUC is 0.7267 for Rankboost_W respectively that show a significant gain over the other prioritization methods. The experiments have proved the performance as we expected that the ensemble approach can outperform the original methods by boosting [2][19].

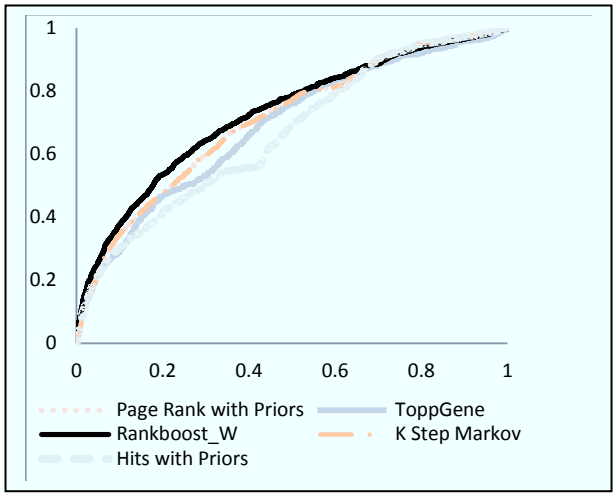

Figure4. *ROC curves for each prioritization method.*

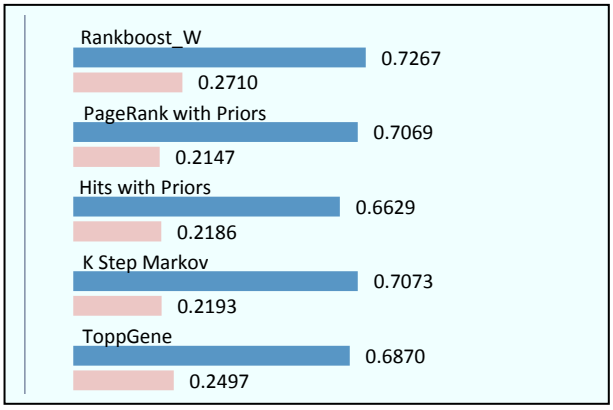

Figure5. *The performance for AUC(dark) and MAP(light)*

IV. Conclusion

We demonstrated that our proposed ranking method had been shown to outperform the previous methods on prostate cancer. The results indicated that integrating various strategies into gene prioritization by ensemble learning was beneficial to the ranking result. This study has taken a step in the direction of integrating each prioritization methods rather than the datasets. The pairwise computation of gene ranking takes in general O(n2) computational complexity. But we use proper setting of threshold for each known gene to reduce unnecessary computation. We ran the experiments on a PC laptop with 2.5 GHz under 64 bits Window 7 OS, the execution time for each known gene prioritization varies from 31 seconds (the best case) to around 12 hours (the worst case). The experiments reported in the present paper have demonstrated that the new architecture can be practically implemented and provide adequate results. The study suggests that ranking methods could be combined into better performance tools in identification of genes relevant to a particular disease. The weighting scheme can be generalized to take care of the condition when number of gene prioritization methods is more than four.

Acknowledgment

The research is partially supported by the Bio-resources Collection and Research Center of Linko Chang Gung Memorial Hospital and National Tsing-Hua University of Taiwan R.O.C. under the grant number 100N2722E1.

References

[1] M. A. Arbib, "Ensemble learning," in The handbook of brain theory and neural networks, ed: The MIT Press, 2003.

[2] R. E. Schapire, The boosting approach to machine learning: An overview vol. 171. New York: Springer, 2003.

[3] S. J. Russell and P. Norvig, Artificial intelligence: a modern approach: Tsinghua University Press, 2006.

[4] P. Yang, Y. Hwa Yang, B. B. Zhou, and A. Y. Zomaya, "A review of ensemble methods in bioinformatics," Current Bioinformatics, vol. 5, p. 296, 2010.

[5] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," The Journal of Machine Learning Research, vol. 4, pp. 933-969, 2003.

[6] L.-C. Tranchevent, F. B. Capdevila, D. Nitsch, B. De Moor, P. De Causmaecker, and Y. Moreau, "A guide to web tools to prioritize candidate genes," Briefings in Bioinformatics, vol. 12, pp. 22-32, Jan 2011.

[7] X. B. Wu, R. Jiang, M. Q. Zhang, and S. Li, "Network-based global inference of human disease genes," Molecular Systems Biology, vol. 4, May 2008.

[8] J. Chen, B. J. Aronow, and A. G. Jegga, "Disease candidate gene identification and prioritization using protein interaction networks," Bmc Bioinformatics, vol. 10, p. 73, 2009.

[9] S. Aerts, D. Lambrechts, S. Maity, P. Van Loo, B. Coessens, F. De Smet, et al., "Gene prioritization through genomic data fusion " , Nature Biotechnology, vol. 24, pp. 537-544, Jun 2006.

[10] D. Nitsch, J. P. Gonçalves, F. Ojeda, B. De Moor, and Y. Moreau, "Candidate gene prioritization by network analysis of differential expression using machine learning approaches," Bmc Bioinformatics, vol. 11, p. 460, 2010.

[11] A. Hamosh, A. F. Scott, J. S. Amberger, C. A. Bocchini, and V. A. McKusick, "Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders," Nucleic Acids Research, vol. 33, pp. D514-D517, 2005.

[12] L. C. Li, H. Zhao, H. Shiina, C. J. Kane, and R. Dahiya, "PGDB: a curated and integrated database of genes related to the prostate," Nucleic Acids Research, vol. 31, pp. 291-293, 2003.

[13] E. W. Sayers, T. Barrett, D. A. Benson, E. Bolton, S. H. Bryant, K. Canese, et al., Database resources of the national center for biotechnology information. Nucleic Acids Research, vol. 39, pp. D38-D51, 2011.

[14] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa, "KEGG: Kyoto encyclopedia of genes and genomes," Nucleic acids research, vol. 27, pp. 29-34, 1999.

[15] R. L. Seal, S. M. Gordon, M. J. Lush, M. W. Wright, and E. A. Bruford, genenames. org: the HGNC resources in 2011. Nucleic Acids Research, vol. 39, pp. D514-D519, 2011.

[16] J. Chen, E. E. Bardes, B. J. Aronow, and A. G. Jegga, "ToppGene Suite for gene list enrichment analysis and candidate gene prioritization," Nucleic acids research, vol. 37, pp. W305-W311, 2009.

[17] N. J. Risch, "Searching for genetic determinants in the new millennium," Nature, vol. 405, pp. 847-856, 2000.

[18] S. Mostafavi, D. Ray, D. Warde-Farley, C. Grouios, and Q. Morris, "GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function," Genome Biol, vol. 9, p. S4, 2008.

[19] B. T. Bartell, G. W. Cottrell, and R. K. Belew, "Automatic combination of multiple ranked retrieval systems," pp. 173-181,1994.
Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," The Journal of Machine Learning Research, vol. 4, pp. 933-969, 2003.

[20] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," journal of computer and system sciences, vol. 55, pp. 119-139, Aug 1997.

[21] A. Masoudi-Nejad, A. Meshkin, B. Haji-Eghrari, G. Bidkhori, " Cadidate gene prioritization", Mol. Genet. Genomics, 287:679-698, 2012.