A gene-based permuted extreme gradient boost method for detecting gene-gene interactions of qualitative trait and application to XXX data

Abstract: Boosted tree is a popular and highly effective method in machine learning for modeling additive models with non-linear terms. In this paper, we propose a novel gene-based, permuted, extreme, gradient boosting method called gpXGB to detect interactions between genes in qualitative traits, which has advantage in both statistical power and biological interpretability. (The main idea is to permute the genotype within each class of the dataset in two ways, one keep the interaction between genes and another remove such interactions, then rank the AUC differences of the result of XGB after these two different types of permutation.)
The framework rank the interacting gene pairs by estimating the AUC difference of a XGB classification model on two test datasets through permutation that one keeping the pairwise interaction while the other removing the interaction.

1. Introduction

Genome-wide association studies (GWAS) have identified over six thousand single-nucleotide polymorphisms (SNPs) associated with complex diseases or traits. Earlier GWAS analysis strategies were largely based on single locus models, which test the association between individual markers and a given phenotype independently. Although this type of approaches have successfully identified many regions of disease susceptibility, most of these SNPs identified have small effect sizes which failed to fully account for the heritability of complex traits. Genetic interaction has been hypothesized to play an important role in the genetic basis of complex diseases and traits and to be one of the possible solutions to this problem of "missing heritability". Even if genetic interaction explains only a tiny fraction of "missing heritability", they can still provide some biological insight on the pathway level through by aiding the construction of novel gene pathway topologies.

The first investigations on genetic interactions have first been investigated at the SNP level, in which various statistical methods, including logic and logistic regression, odds-ratio, linkage disequilibrium(LD) and entropy-based statistic, are employed to detect SNP-SNP interactions (i.e. epistasis). Other techniques that have been used to study SNP-SNP interactions include multifactor dimensionality reduction, Tuning RelieF, Random Jungle, BEAM, BOOST(Wan, Yang et al. 2010) and pRF(Li, Malley et al. 2016). These marker-based methods may encounter some common challenges, such as the complexity arising from the large number of pairwise or higher-order tests because all pairs or groups of SNPs have to be considered; the extensive burden of multiple-testing correction they entail. In this paper, we aim to improve the power of gene-gene interaction detection by moving beyond SNP level, and instead considering all potential pairs of SNPs from each of a pair of genes in a single gene-based interaction detection.

Gene-based test have been proven successful for regular GWAS tests of main (marginal) associations, and there are several potential advantages to extending this methodology to detecting gene-gene interactions. First, a gene-based approach substantially reduces the burden of multiple-testing correction, e.g. for 20,000 genes, there are $\sim 2\times10^{8}$ possible pairwise gene-based interaction tests, while for 3 million SNPs there are over $\sim 5\times10^{12}$ possible marker-based interaction tests. Second, gene-based interaction tests can increase power by aggregating signals across variants in the target regions (a gene or any other locus) when multiple causal interactions influence the phenotype of interest, as has been shown to be case for GWAS tests of main association effect. Third, a gene-based interaction detection is a natural choice when detection is focused on a reduced set of pairs based on prior biological knowledge, which is often on a gene-level, e.g. testing pairs of genes that exhibit protein-protein interactions (PPI) or that participate in the same pathways.
Considering two genes, there may be multiple causal SNP-SNP interactions between them that influence the phenotype of interest. For the reasons above, gene-based gene-gene interaction methods have recently grown in popularity.

In the case-control studies, Peng et al. proposed a U-statistic, called CCU, to measure the difference of correlation between two genes in cases and controls. In CCU(Peng, Zhao et al. 2010), correlations in cases and controls is based on canonical correlation analysis in order to detect gene-gene co-association. Limitation of this methods is only can find linear relationship, which may limit power in the presence of nonlinear correlations between genes. To overcome this limitation, CCU has been extended to KCCU (Yuan, Gao et al. 2012, Larson, Jenkins et al. 2014), where nonlinear correlations provides by applying CCA to kernel-generated feature spaces. Li et al. introduced a new method called GBIGM (Li, Huang et al. 2015), which was based on an entropy-based non-parametric statistic and is a new option to detect non-linear gene-gene interaction. More recently, Emily developed a new method called AGGrEGATOr (Emily 2016), which aim at combining marker-based interaction tests between all pairs of markers in two genes to produce a gene-level test for interaction between the two. Before, the similar strategy has already been successfully used for detection of gene-gene interaction for quantitative phenotype(Ma, Clark et al. 2013).

In this paper, rather than considering design a dedicated statistic, we introduced a machine learning method - extreme gradient boost (Xgboost(Chen and Guestrin 2016)) and proposed a new approach called gene-based permuted extreme gradient boost (gpXGB) to detect gene-gene interaction. It is based on comparing the performance of two test dataset keeping or removing interaction of selected gene pair through permutation strategies on predictive model. The advantage of our new approach for interaction detection, compared with traditional statistical approaches, is that it does not require explicit modeling of interacting terms and allow any kind of the functional form that interaction might take. Statistical methods often represent only multiplicative interactions and thus may miss other nonlinear forms of interactions. When little is known about the trait under study, conducting a fully nonparametric analysis is more flexible for data-driven exploratory genome-wide association studies.

2. Methods

In this section we first detail our gpXGB approach. We then explain the various simulation studies conducted to assess either for the control of type-I error rate or for the power of gene-gene interaction detection. Finally, we present NESDA () dataset used to evaluate our approach in a real situation as well as the gene pairs selected to investigate the capacity for gpXGB to replicate findings.

2.1 gpXGB

In this paper, we propose a machine learning based procedure to detect the interaction between two genes in susceptibility with a binary phenotype, typically a case/control disease status. Let $y \in \{0,1\}$ be the phenotype, where $y = 0$ stands for a control and $y = 1$ a case, and $\mathbf{X}_g$ for $g = 1,...,G$ is a gene with multiple SNP markers, there are totally G genes in the given gene list. Suppose we have a sample of n sample and $\mathbf{Y} = \{y_1,...,y_n\}$ the vector of the observed binary phenotypes. Each gene is a collection of respectively m_i SNPs. The observed genotypes for gene $\mathbf{X}_g$ can be represented by a $n \times m_i$ matrix: $\mathbf{X}_g = [x_{ij}]_{i\in 1,..,n;\, j\in 1,..,m_i}$, where $x_{ij} \in \{0,1,2\}$ is the number of copies of the minor allele for SNP j carried by individual i. We use $\mathbf{X}_g^D$ and $\mathbf{X}_g^C$ denote the genotypes for gene g with disease status as case and control respectively. These matrices may also contain adjusted genotype values that have been corrected for various covariates and population stratification.

Considering the phenotype, it should be an additive model that may have margin effects terms for single SNP markers and linear or nonlinear form of interaction terms for two or more than two SNPs. We choose Xgboost as our classifier because gradient boosting decision tree (GBDT) is a neural way to approximate true target function with additive and non-linear structure and Xgboost is an advanced version for GBDT at precision and computational efficiency.

Our approach has three steps: 1) training 2) permutation 3) testing and ranking. We start by training an Xgboost model with all the genes in the gene list and use cross-validation to choose a best model and save it. Then for each selected pair of genes, we use our permutation strategies to generate two different test datasets, one keep while the other remove the interaction between the selected pair of genes. At last, we calculate the performance difference ΔAUC for the two test datasets on the well-trained model. Each pair of gene has a ΔAUC and we use it as a measurement of strength of gene-gene interaction. The larger ΔAUC, the stronger interaction was indicated for the selected pair of gene. The various steps of the gpXGB framework are illustrated in Figure1.

2.1.1 Xgboost

XGBoost(Chen and Guestrin 2016), a scalable supervised machine learning system for tree boosting, has recently been dominating applied machine learning and Kaggle competitions. It is an advanced version of gradient boosted decision trees (GBDT) with speed and performance improvement.

In this tree ensemble model, the base classifier is CART. The samples were classified into different leaves, and assigned the score on corresponding leaf. A CART is a bit different from decision trees, where the leaf only contains decision values. In CART, a real score is associated with each of the leaves, which gives us richer interpretations that go beyond classification. This also makes the unified optimization step easier.

Depending on different task, such as regression, classification, ranking, etc., the prediction $\hat{y}_i$ given x_i ($x_i \in \mathbf{R}^{(s_1+\ldots+s_g)}$ as $i\text{-}th$ training example) can have different interpretations. Since our data is case-control data, we use logistic regression form:

$$p(y_i = 1 \mid x_i) = \frac{1}{1+\exp(-\hat{y}_i)} \qquad (1)$$

i.e. predict the probability of the sample being case, where $\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F}$, K is the number of trees, f is a function in the functional space $\mathcal{F}$,and $\mathcal{F}$ is the set of all possible CARTs.

Therefore the objective to optimize can be written as:
The objective to optimize can be written as:

$$obj = \sum_{i=1}^{N_D+N_C} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \qquad (2)$$

$l(y_i, \hat{y}_i)$ is the logistic loss function:

$$l(y_i, \hat{y}_i) = y_i \log(1+\exp(-\hat{y}_i)) + (1-y_i)\log(1+\exp(\hat{y}_i)) \qquad (3)$$

$\Omega(f_k)$ is the regularization term.

2.1.1 Additive Training

The parameters of Xgboost are those functions f_i with each containing the structure of tree and the leaf scores. It is not feasible to train all the trees at once because it is hard to take the gradient as the traditional optimization problem do. Instead, Xgboost use an additive strategy: fix the trees have already learned, add one new tree at a time. Note the prediction value at step t by $\hat{y}_i^{(t)}$, so we have

$$
\begin{aligned}
&\hat{y}_i^{(0)} = 0 \\
&\hat{y}_i^{(1)} = f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
&\hat{y}_i^{(2)} = f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\quad \ldots \\
&\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned} \tag{4}
$$

At each step, Xgboost choose the tree that optimizes the objective. Here, the algorithm takes the Taylor expansion of the logistic loss function up to the second order:

$$
obj^{(t)} = \sum_{i=1}^{n} [l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + (1/2) h_i^2 f_t(x_i)] + \Omega(f_t) \tag{5}
$$

Where the $g_i = \dfrac{\partial l(y_i, \hat{y}_i^{(t-1)})}{\hat{y}_i^{(t-1)}}$ and $h_i = \dfrac{\partial^2 l(y_i, \hat{y}_i^{(t-1)})}{\hat{y}_i^{(t-1)}}$

After remove all the constants, the specific objective for the new tree at step t becomes:

$$
obj^{(t)} = \sum_{i=1}^{n} [g_i f_t(x_i) + (1/2) h_i^2 f_t(x_i)] + \Omega(f_t) \tag{6}
$$

This is the optimization goal with loss function as formula (3).

2.1.2 Model Complexity

Let us refine the definition of a tree

$$
f_t(x) = w_{q(x)},\ w \in R^T,\ q : R^d \to \{1, 2, \ldots T\} \tag{7}
$$

Here w is the vector of scores on leaves, q is a function assigning each data point to the corresponding leaf and T is the number of leaves. In XGBoost, the complexity of the tree defines as:

$$
\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \tag{8}
$$

Where the $l2$ norm can smoothing the leaf scores.

2.1.3 The Structure Score

After renormalizing the tree model, we can write the objective value with the t-th tree as:

$$
\begin{aligned}
Obj^{(t)} &\approx \sum_{i=1}^{n} [g_i w_{q(x)} + \frac{1}{2} h_i w_{q(x)}^2] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \\
&= \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T
\end{aligned} \tag{9}
$$

where $I_j = \{i \mid q(x_i) = j\}$ is the set of indices of data points assigned to the j-th leaf. In the second line the index of summation has been changed because all the data points on the same leaf get the same score. It can be further compressed by defining $G_j = \sum_{i \in I_j} g_i$ and $H_j = \sum_{i \in I_j} h_i$:

$$Obj^{(t)} = \sum_{j=1}^{T}[G_j w_j + \frac{1}{2}(H_j + \lambda)w_j^2] + \gamma T \qquad (10)$$

In formula (10), w_j are independent to each other, the form $G_j w_j + \frac{1}{2}(H_j + \lambda)w_j^2$ is quadratic and the best w_j for a given structure $q(x)$ and the best objective reduction is:

$$w_j^* = -\frac{G_j}{H_j + \lambda} \qquad (11)$$

$$Obj^* = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j + \lambda} + \gamma T \qquad (12)$$

The formula (12) measures how good a tree structure $q(x)$ is. This score is like the impurity measure in a decision tree, except that it also takes the model complexity into account.

2.1.4 Learn the tree structure

When come to choose a best tree at step t, enumerating all possible trees is intractable in practice. Xgboost tries to optimize one level of the tree at a time. When splitting a leaf into two leaves, the score it gains is:

$$Gain = \frac{1}{2}[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}] - \gamma \qquad (13)$$

The formula (13) can be decomposed as 1) the score on the new left leaf; 2) the score on the new right leaf; 3) the score on the original leaf; 4) regularization on the additional leaf. If the gain is smaller than γ , it would be better not to add that branch. This is exactly the pruning techniques in tree based models.

XGBoost is developed with both deep consideration in terms of systems optimization and principles in machine learning. In this paper, we choose this algorithm and use grid search for some of the parameter combinations with cross-validation to find the best model.

2.2 Permutation

Previously, the idea of permutation the SNP data was used in Greene's(Greene, Himmelstein et al. 2010) and Jing's(Li, Malley et al. 2016) method. Greene et al. designed an explicit test of epistasis to reflect only nonlinear interaction or epistasis component of the model. They advanced the traditional permutation testing framework by shuffling each column of SNP instead of randomizing the class label, which can generate permuted datasets under the null hypothesis that the only genetic associations in the data are linear or additive in nature and that any nonlinear interaction effects are only there by chance. This yields an explicit test of epistasis when combined with a method such as MDR what is capable of modeling nonlinear interactions. Jing et al. developed a permuted random forest (RF) method which generate two test dataset by permutation, one shuffled SNPs in the selected pair individually, while the other one keep the pair of SNP combination shuffling together. Then use the difference of error rate between the two test dataset on a well-trained RF model to

measure the strength of the interaction of selected SNP pair. Other SNPs except for the selected pair were kept their original form in both permutation strategies, so the interactions among other non-selected SNPs were preserved in both of the permutation framework.

The two methods above are both marker-based. Motivated by them, we designed the gene-based permutation strategy. In the permutation step, for each pair of genes independently, we carried out two permutation strategies to generate two test datasets. The difference between the two test datasets was merely the presentation or deletion of the interaction between the pair of genes. We first sort the data rows (i.e. samples) by class into cases and controls. Then, for the first permutation strategy, we perform the permutation for all the genes $\mathbf{X}_g$ individually in the dataset within each class to remove any interactions between genes in each class. The independent margin effect of each gene was preserved due to the consistent genotype frequencies combination of each gene within each class before and after permutation. Our second permutation strategy aims to only keep the interaction between $\mathbf{X}_i$ and $\mathbf{X}_j$, and independent margin effect for each gene $\mathbf{X}_g$ .In more details, $\mathbf{X}_i$ and $\mathbf{X}_j$ were shuffled together by keeping the combination of them within each class, while other genes $\mathbf{X}_{g\setminus\{i,j\}}$, permute in the dataset within each class, respectively. Compared with Jing's method, XXXXXXXX(emphasize the difference between our permutation and Jing's)

Figure for permutation procedure ***(not finished)***

2.3 Testing and Ranking

We introduce a new approach to gene-based gene-gene interaction detection. It is based on comparing the performance of two test dataset keeping or removing interaction through permutation strategies (in section 2.2) on predictive model (in section 2.1). Our interaction estimation technique is based on the following observation. If $\mathbf{X}_{g1}$ and $\mathbf{X}_{g2}$ interact, then test data of the second permutation strategy have significant better predictive performance than test data of the first permutation strategy, because the latter one cannot reflect the true functional dependency between $\mathbf{X}_{g1}$ and $\mathbf{X}_{g2}$ .On the other hand, if the two gene do not interact, then the absence of the interaction in the test data should not hurts its performance. Hence in the absence of an interaction between $\mathbf{X}_{g1}$ and $\mathbf{X}_{g2}$, the predictive performance on the first test data and the second test data should be comparable.

In this paper, we use AUC (Area Under Curve), the area under the a receiver operating characteristic (ROC) curve, to measure the classifier performance. In machine learning, ROC curve is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is

varied. Consider the ROC plot of the true positive rate vs the false positive rate as the threshold value for classifying an item as 0 or is increased from 0 to 1: if the classifier is very good, the true positive rate will increase quickly and the area under the curve will be close to 1. If the classifier is no better than random guessing, the true positive rate will increase linearly with the false positive rate and the area under the curve will be around 0.5. Generally, the larger AUC is, the better performance the classifier has. Especially, the AUC is independent of the fraction of the test dataset which is class 0 or class 1 that it is useful for evaluating the performance of classifier even on unbalanced data sets.

In the third step of testing, for each pair of genes, both of the permuted datasets were tested using the well-trained Xgboost model to get their AUC scores. Permutation was repeated 100 times and the average AUC was calculated from all permutations. We named the average AUC from the first permutation strategy, AUC1, in which the test dataset only maintain the margin effect of genes in the gene list. And named the average AUC from the second permutation strategy, AUC2, in which the test dataset kept both interaction between selected pair of gene and margin effect of all the genes. Therefore, the subtraction of AUC2 and AUC1 would be the difference of classifier performance caused by the interaction.

In the last step, after each pair of genes was permuted using the two permutation schemes (in the section 2.2) and tested to get the AUC scores, the AUC difference $\Delta AUC = AUC2 - AUC1$ was calculated and used as the measurement of the interaction strength for selected gene pair, since removing an important interaction could have a strong effect on classifier performance. The larger the ΔAUC is, the stronger interaction signal was indicated for that pair of genes. The ΔAUC s were ranked and the pair of genes with the largest ΔAUC having the strongest interaction among all the gene pairs. In practice, we can pick top 5 pairs as the candidate interaction pair or selected candidate pairs through the distribution of ΔAUC .

The ALGORITHM 2 is the framework of gpXGB.

ALGORITHM2: gpXGB

Input: genotype dataset $S = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}, x_i \in R^{(m_1+...+m_g)}, y_i \in \{0,1\}$; gene list file with position information for G genes; buffer region size.

Output: The ranking list sorted by ΔAUC for all the gene pairs in the gene list.

1. Train Xgboost model, using grid search to find the proper parameter combination of the Xgboost, using 5-fold cross validation for each parameter combination and select the best parameter combination that gives the best average predictive performance.
2. XgboostModel = trainXgboost(S , parameter combination)
3. **for** $i = 1$ **to** $G - 1$ **do**
4. **for** $j = i + 1$ **to** G **do**
5. **for** $k = 1$ **to** 100 **do**
6. testData1=TypeIPermu () // generate test data with the first permutation strategy
7. testData2=TypeIIPermu () // generate test data with the second permutation strategy
8. ACU_{k1} =calcAUC(Predict(XgboostModel,testData1))
9. ACU_{k2} = calcAUC(Predict(XgboostModel,testData1))
10. $\Delta AUC_k = ACU_{k2} - ACU_{k1}$ // ΔAUC for the k-th permutation
11. **end for**
12. $\Delta AUC_{ij} = (\sum_{k=1}^{100} \Delta AUC_k) / 100$ //Average ΔAUC for 100 times permutation
13. **end for**
14. **end for**
15. Return C_G^2 pairs of gene interaction ranking list sorted by ΔAUC_{ij} with decrease order

2.4 Simulation study

To evaluate the performance of our gpXGB procedure for gene-based gene-gene interaction detection, we use GAMETES to generate case-control simulation datasets with different parameter setting. Simple description for GAMETES. We simulate two scenarios where the phenotype is as the sum of one pair or multiple pairs (we choose 5) of random, pure and strict 2-locus genetic models. In each sensoria, heritability varied $h \in \{0.005, 0.01, 0.025, 0.05, 0.1, 0.2\}$ and MAF was either 0.2 or 0.4. For each of the 12 genetic constrains combinations, 1,000 models were ranked by CORs and the models with the highest, moderate and lowest EDMs were selected as the three models for data simulation. For each selected model, we simulated 100 replicate datasets under the sample size $n \in \{1000, 2000, 3000, 4000, 5000\}$ with balanced cases and controls. All together, we generated a total of

2.5 Application with protein-protein interactions (PPI) to Netherlands Study of Depression and Anxiety (NESDA) GWAS dataset

3. Results

3.1 methods to compare

KCCU

The kernel canonical correlation-based U-statistic model (KCCU) is a gene-based gene-gene interaction detection method which can reflect nonlinear relationship between two genes in the case-control dataset. In KCCU, for given two genes $l, m \in \{1,...,G\}$, such that $l \neq m$. Consider the genotype matrices $\mathbf{X}_{\mathbf{l}}^{\mathbf{D}}, \mathbf{X}_{\mathbf{m}}^{\mathbf{D}}, \mathbf{X}_{\mathbf{m}}^{\mathbf{C}}$ and $\mathbf{X}_{\mathbf{l}}^{\mathbf{C}}$, with corresponding reduced kernel representations $\mathbf{K}_{\mathbf{l}}^{\mathbf{D}}, \mathbf{K}_{\mathbf{m}}^{\mathbf{D}}, \mathbf{K}_{\mathbf{l}}^{\mathbf{C}}$ and $\mathbf{K}_{\mathbf{m}}^{\mathbf{C}}$. Define $r_{lm}^{D} = CCA(\mathbf{K}_{\mathbf{l}}^{\mathbf{D}}, \mathbf{K}_{\mathbf{m}}^{\mathbf{D}})$ and $r_{lm}^{C} = CCA(\mathbf{K}_{\mathbf{l}}^{\mathbf{C}}, \mathbf{K}_{\mathbf{m}}^{\mathbf{C}})$ to be the respective maximal kernel canonical correlations for case and control between gene l and m. After transform the r_{lm}^{D} and r_{lm}^{C} to an analog of the Fisher's simple correlation coefficient transformation, we can obtain the KCCU statistic. The details of KCCU method can be found in the paper [].

3.2 Simulation studies

3.3 real dataset

4. Discussion

Abstract

Among the large of number of statistical methods that have been proposed to identify gene-gene interactions in case-control genome-wide association studies (GWAS), gene-based methods have recently grown in popularity as they confer advantage in both statistical power and biological interpretation.

Introduction

Sampling of cases and controls was then completed from a sufficiently large number of simulated genotype-phenotype pairs.

For both type I error and power simulations, we consider whether or not explicit marginal effects are included in the disease model. Each simulation scenario is conducted with case-control status sample sizes of 500, 1000, and 1500.

XXX data study

We applied the bagged-pRF approach to detect gene-gene interaction within the XXX pathway, using data from a case-control study of XXX.

Results

Type I error

Power

Application to XXX data

Discussion

XXX data findings

Conflict of interest

Acknowledgements

Gene-based permuted extreme gradient boost (gpXgboost

References

Chen, T. and C. Guestrin (2016). XGBoost: A Scalable Tree Boosting System. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

Emily, M. (2016). "AGGrEGATOr: A Gene-based GEne-Gene interActTiOn test for case-control association studies." Stat Appl Genet Mol Biol **15**(2): 151-171.

Greene, C. S., D. S. Himmelstein, H. H. Nelson, K. T. Kelsey, S. M. Williams, A. S. Andrew, M. R. Karagas and J. H. Moore (2010). "Enabling personal genomics with an explicit test of epistasis." Pac Symp Biocomput: 327-336.

Larson, N. B., G. D. Jenkins, M. C. Larson, R. A. Vierkant, T. A. Sellers, C. M. Phelan, J. M. Schildkraut, R. Sutphen, P. P. Pharoah, S. A. Gayther, N. Wentzensen, C. Ovarian Cancer Association, E. L. Goode and B. L. Fridley (2014). "Kernel canonical correlation analysis for assessing gene-gene interactions and application to ovarian cancer." Eur J Hum Genet **22**(1): 126-131.

Li, J., D. Huang, M. Guo, X. Liu, C. Wang, Z. Teng, R. Zhang, Y. Jiang, H. Lv and L. Wang (2015). "A gene-based information gain method for detecting gene-gene interactions in case-control studies." Eur J Hum Genet **23**(11): 1566-1572.

Li, J., J. D. Malley, A. S. Andrew, M. R. Karagas and J. H. Moore (2016). "Detecting gene-gene interactions using a permutation-based random forest method." BioData Min **9**: 14.

Ma, L., A. G. Clark and A. Keinan (2013). "Gene-based testing of interactions in association studies of quantitative traits." PLoS Genet **9**(2): e1003321.

Peng, Q., J. Zhao and F. Xue (2010). "A gene-based method for detecting gene-gene co-association in a case-control association study." Eur J Hum Genet **18**(5): 582-587.

Wan, X., C. Yang, Q. Yang, H. Xue, X. Fan, N. L. Tang and W. Yu (2010). "BOOST: A fast approach to detecting gene-gene interactions in genome-wide case-control studies." Am J Hum Genet **87**(3): 325-340.

Yuan, Z., Q. Gao, Y. He, X. Zhang, F. Li, J. Zhao and F. Xue (2012). "Detection for gene-gene co-association via kernel canonical correlation analysis." BMC Genet **13**: 83.