The Author(s) *BMC Bioinformatics* 2016, **17**(Suppl 17):476
DOI 10.1186/s12859-016-1334-9

RESEARCH Open Access

DeepGene: an advanced cancer type classifier based on deep learning and somatic point mutations

Yuchen Yuan[1,2†], Yi Shi[2*†], Changyang Li[1], Jinman Kim[1], Weidong Cai[1], Zeguang Han[2] and David Dagan Feng[1,2]

From The 27th International Conference on Genome Informatics
Shanghai, China. 3-5 October 2016

Abstract

Background: With the developments of DNA sequencing technology, large amounts of sequencing data have become available in recent years and provide unprecedented opportunities for advanced association studies between somatic point mutations and cancer types/subtypes, which may contribute to more accurate somatic point mutation based cancer classification (SMCC). However in existing SMCC methods, issues like high data sparsity, small volume of sample size, and the application of simple linear classifiers, are major obstacles in improving the classification performance.

Results: To address the obstacles in existing SMCC studies, we propose DeepGene, an advanced deep neural network (DNN) based classifier, that consists of three steps: firstly, the clustered gene filtering (CGF) concentrates the gene data by mutation occurrence frequency, filtering out the majority of irrelevant genes; secondly, the indexed sparsity reduction (ISR) converts the gene data into indexes of its non-zero elements, thereby significantly suppressing the impact of data sparsity; finally, the data after CGF and ISR is fed into a DNN classifier, which extracts high-level features for accurate classification. Experimental results on our curated TCGA-DeepGene dataset, which is a reformulated subset of the TCGA dataset containing 12 selected types of cancer, show that CGF, ISR and DNN all contribute in improving the overall classification performance. We further compare DeepGene with three widely adopted classifiers and demonstrate that DeepGene has at least 24% performance improvement in terms of testing accuracy.

Conclusions: Based on deep learning and somatic point mutation data, we devise DeepGene, an advanced cancer type classifier, which addresses the obstacles in existing SMCC studies. Experiments indicate that DeepGene outperforms three widely adopted existing classifiers, which is mainly attributed to its deep learning module that is able to extract the high level features between combinatorial somatic point mutations and cancer types.

Background

Cancer is known as a category of disease causing abnormal cell growths or tumors that potentially invade or metastasize to other parts of human body [1]. It has long become one of the major lethal diseases which leads to about 8.2 million, or 14.6%, of all human deaths each year [2].

To alleviate the impact of cancer to human health, considerable research endeavors have been devoted to the related diagnosis and therapy techniques, among which somatic point mutation based cancer classification (SMCC) is an important perspective. The purpose of SMCC is to detect the cancer types or subtypes based on somatic gene mutations from the patient, so that the cancer condition of the patient can be specified. Due to the drop in the cost of DNA sequencing in recent years, the availability of DNA sequencing data has increased dramatically, which greatly promotes the developments of SMCC [3]. Compared with conventional cancer

* Correspondence: yishi@sjtu.edu.cn
†Equal contributors
[2]Key Laboratory of Systems Biomedicine, Shanghai Center for Systems Biomedicine, Shanghai Jiaotong University, Shanghai 200240, China
Full list of author information is available at the end of the article

classification methods that are mostly based on morphological appearances or gene expressions of the tumor, SMCC is particularly effective in differentiating tumors with similar histopathological appearances [4] and is significantly more robust to environmental influences, thus is favorable in delivering more accurate classification results. Other genetic aberrations such as copy number variance, translocation, and small insertion or deletion have also been shown to be associated with different cancers [5, 6], but due to the major causal role of somatic point mutations and potential application consideration, we only focus on this kind of genetic aberration in this study. Moreover, the combinatorial point mutation patterns learned in predicting cancer types/subtypes can be used for developing diagnostic gene marker panels that are cost effective. This is particularly true , when compared to DNA amplifications and rearrangements which usually require whole genome sequencing and is expensive for patients, especially regarding time series and whole genome sequencing used in tracing tumor linage evolution during cancer progression.

Clinically, SMCC may significantly facilitate cancer-related diagnoses and treatments, such as personalized tumor medicine [7], targeted tumor therapy [8] and compound medicine [9]. It can also aid cancer early diagnosis (CED) in combination with the sampling and sequencing of circulating tumor cells (CTCs) or circulating DNA (ctDNA) [10–12]. Given the promising applications above, SMCC is widely studied in recent researches [13–15].

In recent years, the drastic developments of machine learning methods have greatly facilitated the researches in bioinformatics, including SMCC. In order to predict the cancer types/subtypes more effectively, many machine learning approaches have been applied in existing cancer type prediction works, which have shown promising results [16–18]. Currently, remarkable developments have been demonstrated in tumor cases of colorectal [19], breast [20], ovary [21], brain [22], and melanoma [23]. However, there are at least three major unresolved challenges:

(1) Normal sequencing results involve extremely large number of genes, usually in tens of thousands, but only a small discriminatory subset of genes is related to the cancer classification task. The other genes are largely irrelevant genes whose existence will only obstruct the cancer classification. Many recent works have been conducted in identifying the discriminatory subset of genes. For example, Cho et al. [24] apply the mean and standard deviation of the distances from each sample to the class center as criteria for classification; Yang et al. [25] improve the method in [24] and bring inter-class variations into the algorithm; Cai et al. [26] propose the clustered gene selection, which groups the genes via *k*-means clustering and picks up the top genes in each group that are closest to the centroid locations. These methods are simple and effective in some cases, but their heuristics are designed for continuous gene expression data, and are not directly applicable to discrete, and especially binary point mutation data.

(2) Even within the discriminatory subset, the majority of genes are not guaranteed to contain informative point mutations and often remain normal (i.e. zero values in the data) [27], which results in extremely sparse gene data (even all-zeros) that is difficult to classify. Yet, to the best of our knowledge, there has been no existing work specifically devised for reducing the data sparsity for SMCC.

(3) Different genes related to specific types of cancer are generally correlated and have complex interactions which may impede the application of conventional simple linear classifiers such as linear kernel support vector machine (SVM) [28]. Therefore, an advanced classifier being capable of extracting the high level features within the discriminatory subset is desired. Although there have been recent works utilizing sparse-coding [29] or auto-encoder [17] for gene annotation, no work has been devoted in applying high-level machine learning approaches to SMCC.

In recent years, the developments of deep neural network (DNN) [30] have equipped bioinformaticians with powerful machine learning tools. DNN is a type of artificial neural network that aims to model abstracted high-level data features using multiple nonlinear and complex processing layers, and provides feedback via back-propagation [31]. First introduced in 1989 [32], DNN has garnered tremendous developments and is widely applied in image classification [33, 34], object localization [35, 36], facial recognition [37, 38], etc. DNN has the potential to introduce novel opportunities for SMCC where it perfectly fits the need for large scale data processing and high level feature extraction. However, to the present, applying customized DNN on SMCC is yet to be explored.

In this paper, we propose a novel SMCC method, named DeepGene, designed to simultaneously address the three identified issues. DeepGene is a DNN-based classification model composed of three steps. It first conducts two pre-processing techniques, including the clustered gene filtering (CGF) based on mutation occurrence frequency, and the indexed sparsity reduction (ISR) based on indexes of non-zero elements; the gene data is then classified by a fully-connected DNN classifier into a specific cancer type. The proposed DeepGene model has four distinct contributions:

(1) The proposed CGF procedure locates the discriminatory gene subset based on mutation occurrence frequency. CGF utilizes features from the whole dataset instead of the current sample alone (e.g. mean and standard deviation), and thus more objectively reflects the correlations among the genes which can more effectively summarize the discriminatory subset. In addition, CGF does not require any prior knowledge from the original data and therefore functions well on both discrete and binary point mutation data.

(2) The proposed ISR procedure converts the sparse gene data into indexes of its non-zero elements. ISR eliminates the vast majority of zero gene elements, and significantly reduces the complexity of the gene data during such process.

(3) We establish a fully connected DNN classifier that uses the gene data after CGF and ISR for cancer classification. With the capacity of high-level feature extraction, our classifier is able to effectively extract deep features from the complexly correlated gene data, and significantly improve the classification accuracy compared with conventional simple linear classifiers such as SVM.

(4) We compile and release the TCGA-DeepGene dataset, which is a reformulated subset of the widely applied TCGA dataset [39] in genome-related researches. TCGA-DeepGene selects 22,834 genes of 12 types of cancer from 3122 different samples, and regularizes the data in a unified format so that classification tasks can be readily performed.

The flowchart of DeepGene is shown in Fig. 1. We conduct experiments on the proposed TCGA-DeepGene dataset, and DeepGene is evaluated against three widely adopted classification methods for SMCC. The results demonstrate that DeepGene has generated significantly higher performance in terms of testing accuracy against the comparison methods.

Methods

DeepGene has three major steps, namely clustered gene filtering (CGF), indexed sparsity reduction (ISR), and DNN-based classification. The CGF and ISR are two independent pre-processing modules, the results of which are then concatenated in the final DNN classifier.

Clustered gene filtering

The CGF step is based on the mutation occurrence frequency of the gene data, and its workflow is summarized in Table 1. Let $A \in \{0, 1\}^{m \times n}$ be the matrix of raw data with binary value, where the n columns correspond to the n samples (cases) in the dataset, and the m rows correspond to the m genes per sample. The binary value indicates whether a mutation is observed:

$$\mathrm{A}(i,j) = \begin{cases} 1 \text{ if mutation obsereved at gene i of sample j} \\ 0 \qquad\qquad\qquad \text{otherwise} \end{cases}.$$

We first sum A by row, and concatenate the result with the row indexes for later reference (step 1 in Table 1):

$$A_{sum} = \begin{bmatrix} 1 & \\ \vdots & sum(A, axis = row) \\ m & \end{bmatrix}.$$

Since the genes with higher occurrence frequency are of more interest, the rows of A_{sum} are sorted in descending order by the second column as A^*_{sum}. After that, we only keep its index column:

$$A^*_{sum} = A^*_{sum}(:, 1).$$

The next step is to group A^*_{sum} by inter-gene similarity (step 3 in Table 1). For two $1 \times n$ gene samples p and q, we use the Jaccard coefficient as their inter-sample similarity $d(p, q)$:

$$d(p, q) = \frac{sum(pq)}{sum(p|q)},$$

where "&" and "|" stand for logical AND and OR.

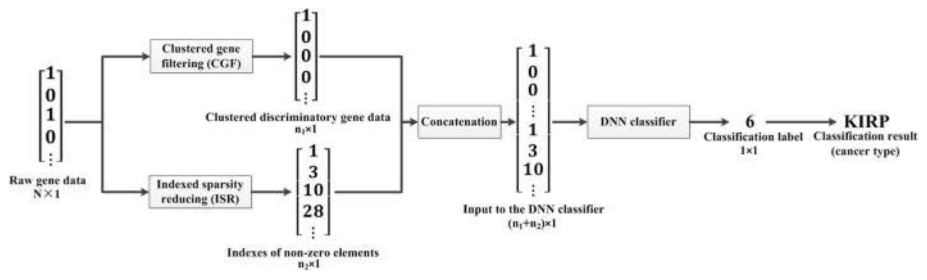
Clustered gene filtering (CGF)
Clustered discriminatory gene data
n₁×1
Raw gene data
N×1
Indexed sparsity reducing (ISR)
Indexes of non-zero elements
n₂×1
Concatenation
Input to the DNN classifier
(n₁+n₂)×1
DNN classifier
6
Classification label
1×1
KIRP
Classification result
(cancer type)

Fig. 1 Flowchart of the proposed DeepGene method. The raw gene data is first pre-processed by the clustered gene filtering (CGF) and the indexed sparsity reduction (ISR), respectively, and then fed into the DNN classifier. The output label from the DNN indicates the cancer type of the input gene sample

Table 1 Workflow of Clustered Gene Filtering (CGF)

```
Input: Gene data matrix $A \in \{0,1\}^{m \times n}$, distance threshold $d_{CGF}$, group element threshold $n_{CGF}$.
1: Sum $A$ by row and sort the result in descent order, and then obtain the sorted index $A^*_{sum}$;
2: Initialize each element as ungrouped;
3: For each ungrouped element $i$ in $A^*_{sum}$:
   (a) For each ungrouped element $j$ in $A^*_{sum}$ other than $i$:
      i. Calculate the similarity $d(A(i,:), A(j,:))$;
      ii. If $d(A(i,:), A(j,:)) > d_{CGF}$, assign $j$ into the group of $i$;
4: Set the output gene index set $g_{out} = \varnothing$;
5: For each group $c$ of $A$ after step 3:
   (a) If group element number $n_c \geq n_{CGF}$, select the top $n_{CGF}$ genes with the highest mutation occurrence frequency as $g_c$;
   (b) $g_{out} = g_{out} \cup g_c$;
6: Apply the index set $g_{out}$ on $A$ and get the filtered gene data $A_{CGF} = A(g_{out}, :)$;
Output: $A_{CGF}$, i.e. the gene data after CGF
```

Starting from $A^*_{sum}(1)$, which stands for the index of the gene with the highest occurrence frequency, we calculate its similarity with each of the following genes. If their similarity is larger than a predefined threshold d_{CGF}, the latter gene is merged into the group of $A^*_{sum}(1)$. After the loop for $A^*_{sum}(1)$, we conduct the loop for the next ungrouped element in A^*_{sum}, until all the genes are grouped with a unique group ID.

The final step is to filter the elements from each group and form the discriminatory subset. We do this by selecting the top n_{CGF} genes in each group with the highest mutation occurrence frequency, where n_{CGF} is another predefined threshold. Groups that have fewer than n_{CGF} elements are discarded. All of the selected genes are then united as the result of CGF (steps 5 and 6 in Table 1).

Indexed sparsity reduction

Although the CGF can effectively locate the discriminatory gene subset and filter out the majority of irrelevant genes, it is still probable that the selected gene subset being highly sparse, i.e. most of the elements in A_{CGF} are zeros. The high sparsity is likely to obscure any distinguishable feature in the gene data and severely hinder the classification. Hence, an effective process in reducing the gene data sparsity is highly desired.

To address the data sparsity issue, we propose the indexed sparsity reduction (ISR) procedure, which minifies the sparsity by converting the gene data into the indexes of its non-zero genes. For a $1 \times n$ gene sample $p \in \{0,1\}^{1 \times n}$, let the number of its non-zero element be n_{NZ}. We set a pre-defined threshold n_{ISR}. If $n_{NZ} \geq n_{ISR}$, find the indexes of its top n_{ISR} non-zero elements that have the highest occurrence frequency in A^*_{sum} of the previous section, and these n_{ISR} indexes are listed in ascending order as a vector p_{ISR}, which is the output of ISR; if $n_{NZ} < n_{ISR}$, we conduct zero-padding to the tail of p_{ISR} to make it has the length of n_{ISR}. The workflow of ISR is illustrated in Fig. 2.

The significance of ISR is apparent. For each gene sample p, ISR filters out the majority of its zero elements and leaves most (if $n_{NZ} \geq n_{ISR}$) or all (if $n_{NZ} < n_{ISR}$) of its non-zero elements. Since $n_{ISR} \ll length(p)$, the percentage of zero elements will drop dramatically after ISR, which means the impact of data sparsity will be significantly suppressed.

DNN-based classifier

As introduced in the previous two sections, both CGF and ISR have their own advantages when conducted alone. However, the performance can be even higher if they are combined together (see more details in the "Evaluate the effect of combining CGF and ISR" Section). We thus combine both CGF and ISR as the preprocessing for our DNN-based classifier.

As shown in Fig. 1, the raw gene data is processed by CGF and ISR, separately, and then concatenated as the input of the DNN classifier. The concatenation is conducted by appending the output of ISR to the tail of the output of CGF, by which the two outputs form a new and longer data vector. The classifier is a feed-forward artificial neural network with fixed input and output size, and multiple hidden layers for data processing. For a hidden layer l, its activation (or output value to the next layer) is computed as:

$$x_l = f(z_{l-1}),$$

where f is the activation function, z_l is the total weighted sum of the input:

$$z_l = W_l x_l + b_l,$$

where W_l and b_l are the weight matrix and bias vector of layer l (to be learned in training). In our case, we adopt the ReLU [40] function as f, and x_1 is the input gene data after pre-processing. The size of the last layer L's output x_L equals to the number of cancer types n_{cancer} ($n_{cancer} = 12$ in our case). x_L is then processed by a softmax layer [41], and the loss J is computed by the logarithm loss function:

$$J = -\sum_{i=1}^{n_{cancer}} y_i \log P_i,$$

where $y_i \in \{0,1\}$ is the ground truth label of cancer type i, and

$$P_i = \frac{x_L(i)}{\sum_j \exp(x_L(j))}$$

is the softmax probability of cancer type i.

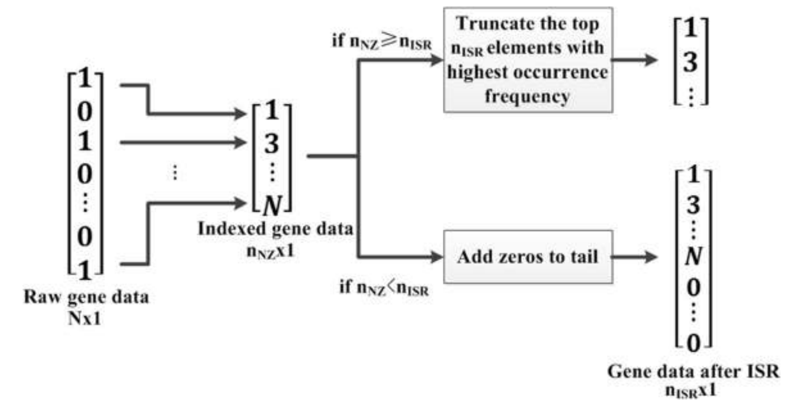
if $n_{NZ} \geq n_{ISR}$
Truncate the top n_{ISR} elements with highest occurrence frequency
if $n_{NZ} < n_{ISR}$
Add zeros to tail
Raw gene data Nx1
Indexed gene data n_{NZ}x1
Gene data after ISR n_{ISR}x1

Fig. 2 Flowchart of the Indexed Sparsity Reduction (ISR) step. After indexing of the non-zero elements, if $n_{NZ} \geq n_{ISR}$, select the top n_{ISR} non-zero elements that have the highest occurrence frequency; if $n_{NZ} < n_{ISR}$, we conduct zero-padding to the tail of the output data so that it has the length of n_{ISR}

In training, the loss J is transferred from the last layer to the former layers via back-propagation [32], by which the parameters W and b of each layer are updated. The training then enters the next epoch, and the feed-forwarding and back-propagation are conducted again. The training stops when a pre-defined epoch number is reached. In testing, only the feed-forwarding is conducted (for once) for a testing sample, and the type of cancer i corresponding to the largest softmax probability of P_i is adopted as the classification result. The workflow of the DNN classifier is summarized in Table 2, and the complete flowchart of DeepGene is illustrated in Fig. 1.

Results

Experiment setup

Dataset

Our experiments are all conducted on the newly proposed TCGA-DeepGene dataset, which is a re-formulated subset of The Cancer Genome Atlas (TCGA) dataset [39] that is widely applied in genomic researches.

The TCGA-DeepGene subset is formulated by assembling the genes that contain somatic point mutation on each of the 12 selected types of cancer. Detailed sample and point mutation statistics for each cancer type can be found in Table 3. The data is collected from the TCGA database with filter criteria IlluminaGA_DNASeq_Curated updated before April, 2015. The mutation information for a gene is represented by a binary value according to one or more mutation(s) (1) or without mutation (0) on that gene for a specific sample. We assemble a total of 22,834 genes from the 3122 samples, and generate a $22,834 \times 3,122$ binary data matrix (i.e. the original data matrix A). This data matrix is the product of our proposed TCGA-DeepGene subset, where each sample (column) is assigned one of the labels {1, 2, ..., 12} meaning the 12 types of cancer above.

To facilitate the 10-fold cross validation in the following experiments, we randomly divide the samples in each of the 12 cancer categories into 10 subgroups, and each time we union one subgroup from each cancer category as the validation set, while all the other subgroups are combined as the training set. This formulates 10 training/validation configurations with fair distributions of the 12 types of cancer, and will be used for the 10-fold cross validation in our following experiments.

Table 2 Workflow of DNN classifier

```
Input: Gene data matrix $A \in \{0, 1\}^{m \times n}$ after CGF and ISR, where rows
and columns correspond to samples and genes, respectively; max
training epoch $E_{\max}$.
1: Training: for each training epoch $e \leq E_{\max}$:
  (a) For each sample $a_i = A(i, :)$:
      i. Conduct feed-forwarding and compute the loss $J$;
      ii. Conduct back-propagation to update the $W$ and $b$
2: Testing: for each sample $a_i = A(i, :)$:
  (a) Conduct feed-forwarding and get softmax probability $P$;
  (b) Adopt the cancer type correspond to max($P$) as the result of $a_i$.
Output: Trained network model (training) or classification results for the
samples (testing).
```

Constant parameters

For the proposed DNN classifier, the output size is set to 12 (i.e. the 12 types of cancer to be classified); the total training epoch $E_{\max}$ is set to 50; the learning rate is set to 50-point logarithm space between 10^{-1} and 10^{-4}; the weight decay is set to 0.0005; and the training batch size (i.e. the number of samples per training batch) is set to 256.

Additionally, in order to facilitate the evaluation of variable parameters, we set each parameter a default value: the distance threshold is set to 0.7; the group element threshold n_{CGF} is set to 5; the non-zero element threshold n_{ISR} is set to 800; the hidden layer number and parameters per layer of the DNN classifier are set to 4 and 8192, respectively.

Evaluation metrics

For all the evaluations in our experiments, we randomly select 90% (2810) samples for training, and the rest 10% (312)

Table 3 Sample and mutation statistics of the TCGA-DeepGene dataset on 12 cancer types

Cancer name	Sample number	Missense mutation	Nonsense mutation	Nonstop mutation	RNA	Silent	Splice_Site	Translation start site	Total mutation
ACC	91	6741	501	15	368	2534	344	42	10,545
BLCA	130	24,067	2142	46	0	9662	528	55	36,500
BRCA	992	55,063	4841	133	3998	17,901	1424	0	83,360
CESC	194	26,606	2716	84	5595	9765	527	0	45,293
HNSC	279	31,416	2545	44	0	12,149	776	0	46,930
KIRP	171	8910	499	17	394	3411	524	0	13,755
LGG	284	5341	378	7	102	2074	294	0	8196
LUAD	230	44,800	3477	46	0	15,594	1377	99	65,393
PAAD	146	21,067	1496	19	859	7936	1005	111	32,493
PRAD	261	9628	563	15	652	3750	513	55	15,176
STAD	288	82,265	4200	92	48	33,344	1868	227	122,044
UCS	56	3070	187	2	234	1114	171	0	4778
Total	3122	318,974	23,545	520	12,250	119,234	9351	589	484,463

samples for testing. In parameter optimization steps for DeepGene, we adopt the 10-fold cross validation accuracy on the training set as the evaluation metric; on the other hand, in the comparison with widely adopted models, we adopt the testing accuracy as the evaluation metric.

Implementation

The CGF and ISR steps are implemented by original coding in MATLAB, while the DNN classifier is implemented on the MatConvNet toolbox [42], which is a MATLAB-based convolutional neural network (CNN) toolbox with various extensibilities.

Evaluation of design options

Determination of CGF's variable

There are two variables that need to be experimentally determined for the CGF step, namely the distance threshold d_{CGF} and the group element threshold n_{CGF}.

Table 4 10-fold cross validation accuracies (%) of DeepGene with different n_{CGF} (row) and d_{CGF} (column)

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
1	38.9	48.0	49.8	48.5	50.6	50.9	53.5	50.4	49.2
2	41.0	51.4	52.7	51.8	52.2	53.7	56.4	53.7	53.2
3	48.1	50.6	51.9	52.3	51.3	51.4	55.6	52.3	51.7
4	51.4	55.3	56.6	56.5	56.0	56.4	60.3	57.1	55.5
5	48.3	58.9	60.2	59.4	59.8	59.5	**63.9**	60.2	59.9
6	48.1	56.7	58.0	56.8	58.6	59.2	61.7	57.9	58.4
7	48.8	54.9	56.2	55.6	55.7	56.5	59.9	57.0	55.5
8	43.9	52.9	54.2	53.0	54.3	54.7	57.9	54.3	54.3
9	41.8	50.5	51.8	50.6	51.0	52.5	55.5	51.7	51.3
10	41.5	44.7	46.0	45.7	45.0	47.1	49.7	45.8	46.4

The optimal result is marked in red. Mean accuracy: 53.0%; standard deviation: 5.01%; maximum accuracy: 63.9%; minimum accuracy: 38.9%. The corresponding 3D bar-plot is shown in Fig. 3a for sensitivity review

To determine the two variables, we change them in 2-dimensional manner, while keeping all the other variables the default values as described in the “Constant parameters” Section. The corresponding 10-fold cross validation accuracies are listed in Table 4, and the corresponding 3D bar-plot to present sensitivity is shown in Fig. 3a. We adopt $d_{CGF} = 0.7$ and $n_{CGF} = 5$ for the following experiments based on the observed experimental results, since they contribute to the optimal performance.

Determination of ISR’s variable

The non-zero element threshold n_{ISR} needs to be experimentally determined for the ISR step. We monitor the number of non-zero elements for each sample in the dataset, and plot the corresponding histogram in Fig. 4. It is seen that 3030 (or more than 97%) of the 3122 samples have less than 800 non-zero genes among the total 22,834 genes. We thus adopt $n_{ISR} = 800$, which not only concentrates the data to the non-zero elements, but also greatly shrinks the data length.

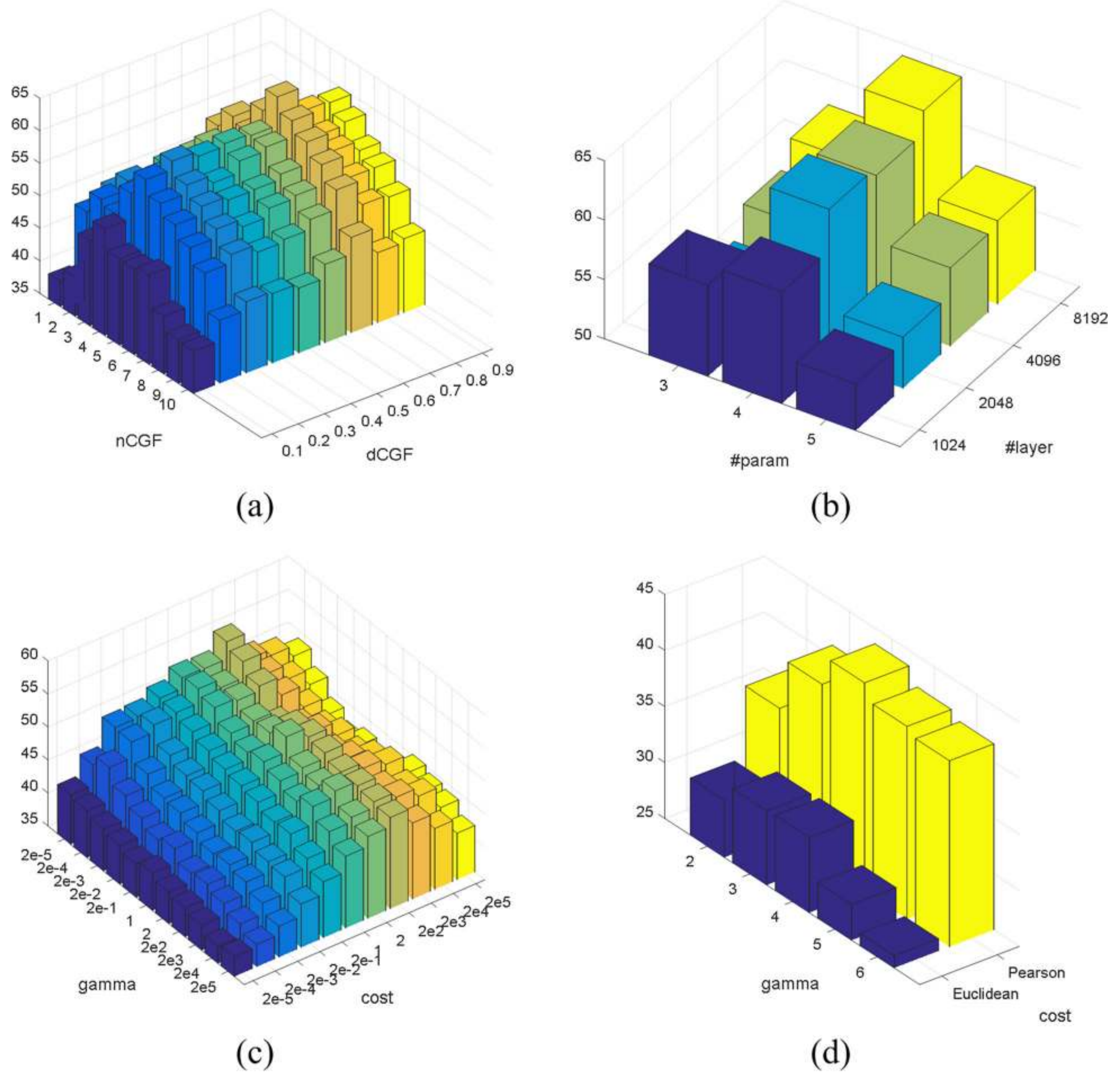

Fig. 3 3D bar-plots of parameter estimations for sensitivity review. The Z-axis stands for 10-fold cross validation accuracy. **a** Parameter estimation for d_{CGF} and n_{CGF}, corresponding to Table *4*; **b** parameter estimation for layer number and parameter number per layer for the DNN classifier, corresponding to Table *5*; **c** parameter estimation for cost and gamma for SVM, corresponding to Table *6*; **d** parameter estimation for Table *7*

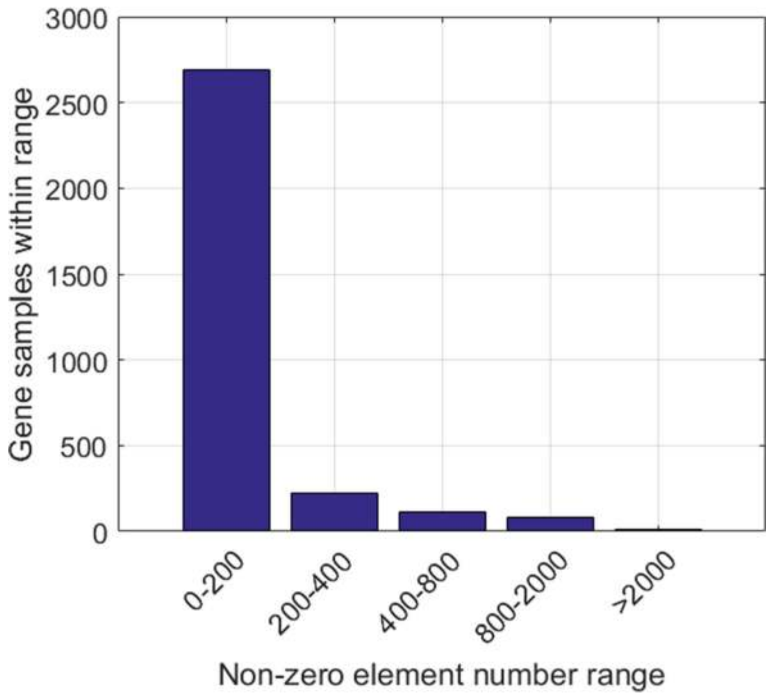

Fig. 4 Non-zero element distribution of the gene samples in the TCGA-DeepGene dataset. Ninety-seven percent of all the 3122 samples have no more than 800 non-zero gene elements

Determine the network architecture

We also need to determine the network architecture for the DNN classifier, which involves two variables: the hidden layer number (#layer) and the parameter number per layer (#param). Enlightened by [43], we monitor the classifier's 10-fold cross validation accuracy with various hidden layer numbers and parameter numbers, the results of which are listed in Table 5, and the corresponding 3D bar-plot to present sensitivity is shown in Fig. 3b. We see that the performance reaches optimal at #layer = 4 and #param = 8192. These values are thus adopted in our following experiments.

Evaluate the effect of combining CGF and ISR

After determining the related parameters for the three steps of DeepGene, we evaluate the impact of our two major innovations, i.e. CGF and ISR. It is mentionable that we conduct CGF and ISR separately and concatenate their results (as shown in Fig. 1) instead of conducting them consecutively. The reason is that the outputs of CGF and ISR are binary data and index data, respectively. Consecutive conduction will only leave the index data (from ISR), while separate conduction can benefit from both the binary data and the index data, thus introduces less bias.

Based on Fig. 1, we compare the performances of the DNN classifier with different configurations:

(1) CGF and ISR (i.e. the proposed input structure);
(2) Only CGF (the upper half of Fig. 1);
(3) Only ISR (the lower half of Fig. 1);
(4) Neither CGF nor ISR (use the raw gene data instead).

Table 5 10-fold cross validation accuracies (%) of DeepGene with different #layer (row) and #param (column)

	1024	**2048**	**4096**	**8192**
3	57.7	53.2	55.6	57.6
4	59.4	62.8	62.1	**64.0**
5	53.9	54.3	56.6	57.0

The optimal result is marked in red. Mean accuracy: 57.9%; standard deviation: 3.42%; maximum accuracy: 64.0%; minimum accuracy: 53.2%. The corresponding 3D bar-plot is shown in Fig. 3b for sensitivity review

The 10-fold cross validation results are shown in Fig. 5. It is clearly observed that the complete CGF + ISR outperforms both CGF and ISR when conducted alone, and also significantly outperforms the raw data without any pre-processing.

Comparison with widely adopted models

We then select three most representative data classifiers that are commonly used in SMCC as comparison methods, namely Support Vector Machine (SVM) [28], *k*-Nearest Neighbors (KNN) [44] and Naïve Bayes (NB) [45]. In order to exhibit the pre-processing effect of CGF and ISR, all the comparison methods use raw gene data as inputs. The three methods are set up as below.

SVM: we use the LIBSVM toolbox [46] in implementing the SVM. Based on the results of a previous work for gene classification [26], the kernel type (−t) is set to 0 (linear kernel). Note that due to the feature set is high dimensional, the linear kernel is suggested over the RBF (Gaussian) kernel [46]; this suggestion is consistent to our trial and error experience on this problem. A 10-fold cross validation is conducted to optimize the parameters cost (−c) and gamma (−g), and the other parameters are set as their default values. The cross validation results are shown in Table 6, and the corresponding 3D bar-plot to present sensitivity is shown in Fig. 3c. We adopt $2^2 = 4$ and $2^{-5} = 0.0313$ for -c and -g, respectively, which lead to the best results in Table 6.

KNN: we compare the performances of Euclidean distance and Pearson correlation coefficient, which are the two most commonly used similarity measures in gene data analysis [26]. The 10-fold cross validation results of the two similarity measures with different neighborhood numbers are shown in Table 7, and the corresponding 3D bar-plot to present sensitivity is shown in Fig. 3d. We adopt the Pearson correlation coefficient and set the neighborhood number to 4, which lead to the optimal validation accuracy.

NB: following [47], the average percentage of non-zero elements in the samples of each cancer category is set as the prior probability.

In the performance comparison between different models, the testing accuracy is adopted as the evaluation metric (see the “Evaluation metrics” Section), which is generally slightly lower than the 10-fold validation accuracy of the corresponding model. The experiment results are plotted in Fig. 6. DeepGene shows significant advantage against all the three comparison methods. The performance improvements are 24.3% (65.5% vs. 52.7%), 60.5% (65.5% vs. 40.8%) and 710% (65.5% vs. 9.23%) against SVM, KNN and NB, respectively. To further validate the performance of the DNN classifier itself without CGF and ISR, we also record the accuracy of the DNN classifier with raw gene data, which is the same input as the comparison methods. The results are shown in Fig. 7, in which the DNN classifier still has the

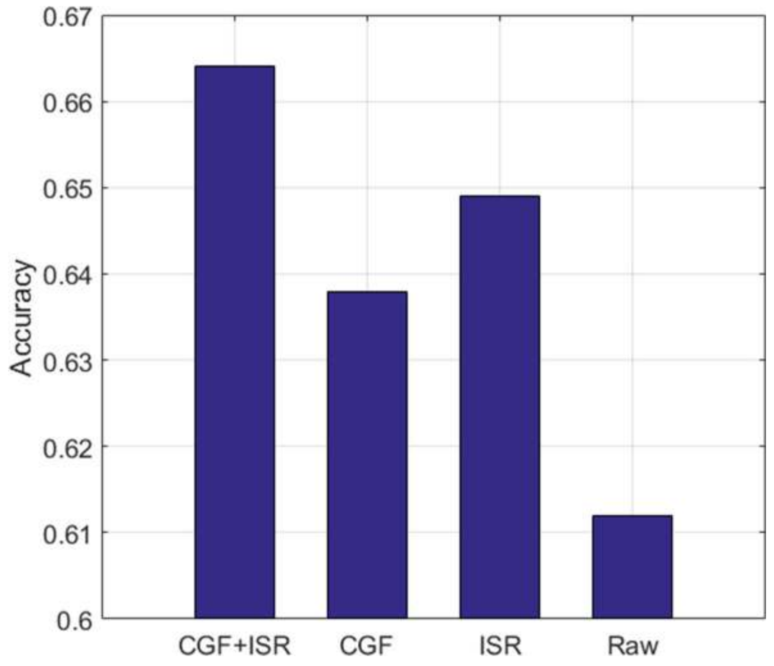

Fig. 5 10-fold cross validation accuracy of DeepGene with different design options. Performance comparison of the complete DeepGene input structure (CGF + ISR), CGF only, ISR only and raw gene data. The complete DeepGene shows significant advantage against the other three options

Table 6 10-fold cross validation accuracy (%) of SVM with different cost (row) and gamma (column) parameters

	2^{-5}	2^{-4}	2^{-3}	2^{-2}	2^{-1}	2^{0}	2^{1}	2^{2}	2^{3}	2^{4}	2^{5}
2^{-5}	42.5	45.6	49.8	50.0	51.3	52.9	52.8	**55.4**	51.3	50.4	48.8
2^{-4}	42.1	47.2	49.6	50.7	51.1	53.2	51.9	54.4	51.0	48.8	46.5
2^{-3}	41.3	44.9	46.7	48.1	50.2	52.4	52.0	52.7	49.9	46.6	43.9
2^{-2}	40.8	43.5	45.1	47.0	49.3	50.5	50.9	48.9	47.1	45.4	42.3
2^{-1}	40.1	42.2	44.8	46.7	48.5	49.6	51.1	50.4	48.4	46.3	43.3
2^{0}	40.7	42.9	43.8	45.7	48.0	50.2	52.5	51.0	47.2	45.5	42.1
2^{1}	39.8	41.2	42.7	44.6	47.5	49.3	49.8	50.6	48.3	46.9	43.1
2^{2}	39.4	41.5	43.0	44.9	46.5	48.2	49.1	49.7	48.8	47.9	45.2
2^{3}	38.8	40.6	42.4	44.0	46.7	49.4	50.3	49.6	47.7	46.4	44.4
2^{4}	37.6	39.5	41.2	43.7	45.0	47.6	48.3	49.2	48.4	47.1	43.9
2^{5}	38.0	38.4	39.7	41.6	43.5	45.9	47.4	48.8	46.5	44.2	42.0

The optimal result is marked in red. Mean accuracy: 46.6%; standard deviation: 3.97%; maximum accuracy: 55.4%; minimum accuracy: 37.6%. The corresponding 3D bar-plot is shown in Fig. 3c for sensitivity review

optimal accuracy (60.1%) against all of the comparison methods.

Discussion

Clustered gene filtering

The main purpose of the CGF step is to filter out irrelevant genes in the samples and locate the candidate discriminatory gene subset. It first groups the genes based on popularity (mutation occurrence frequency) and inter-sample similarity, and then selects the top genes in each group, and finally unites all the genes selected as the output.

The two required parameters, d_{CGF} and n_{CGF}, are experimentally determined (as shown in Table 4). The two adopted values, $d_{CGF} = 0.7$ and $n_{CGF} = 5$, are in the midstream of the evaluation ranges, which are more reliable than the marginal values.

By comparing the performances of the CGF against raw gene data, as the second and fourth bars in Fig. 5 indicate, the CGF has exhibited significant performance boosting. It raises the validation accuracy by 4.25% (from 61.2 to 63.8%), and also contributes to the high performance of the combined CGF + ISR input structure. The advantage of CGF lies in its ability to mask out the majority of irrelevant genes, thus maximally suppress their negative influence, and only focus the data to the discriminatory gene subset.

Indexed sparsity reduction

The ISR step is meant to reduce the data sparsity by converting the gene data into the indexes of its non-zero elements. In that case only the non-zero elements' information is left, while all the zero elements are discarded. The data sparsity will thus be tremendously reduced, making the subsequent classifier only focus on the informative non-zero elements.

The required parameter n_{ISR} is experimentally determined. We monitor the non-zero element distribution among all of the 3122 samples in the TCGA-DeepGene dataset, and record the non-zero element range of each sample. Figure 4 indicates that 97% of the samples have no more than 800 non-zero elements (which are only 3.5% of the total 22,834 genes per sample). We thus set $n_{ISR} = 800$, which is able to reduce the majority of the data sparsity while maximally reserving the discriminatory information of the samples.

Like CGF, ISR has exhibited significant contribution in improving the performance of our classifier, as the third bar in Fig. 5 indicates. It raises the accuracy against raw gene data by 6.05% (from 61.2 to 64.9%), which is even

Table 7 10-fold cross validation accuracies (%) of KNN with different similarity measures (row) and neighborhood numbers (column)

	2	3	4	5	6
Euclidean distance	30.1	31.6	31.7	28.2	26.1
Pearson correlation coefficient	36.4	41.0	**43.6**	42.2	41.6

The optimal result is marked in red. Mean accuracy: 35.3%; standard deviation: 5.63%; maximum accuracy: 43.6%; minimum accuracy: 28.2%. The corresponding 3D bar-plot is shown in Fig. 3d for sensitivity review

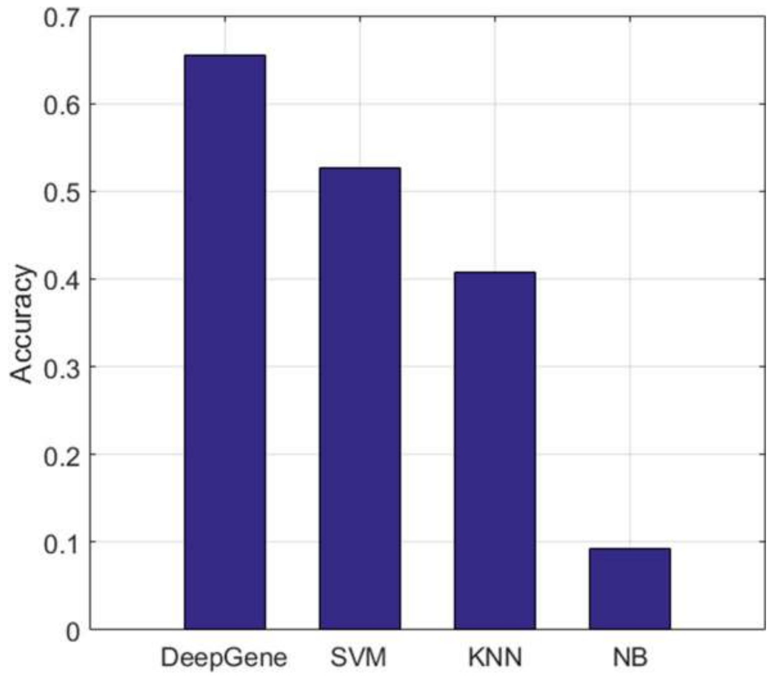

Fig. 6 Testing accuracy of DeepGene against three widely adopted classifiers. DeepGene is clearly advantageous to the comparison methods

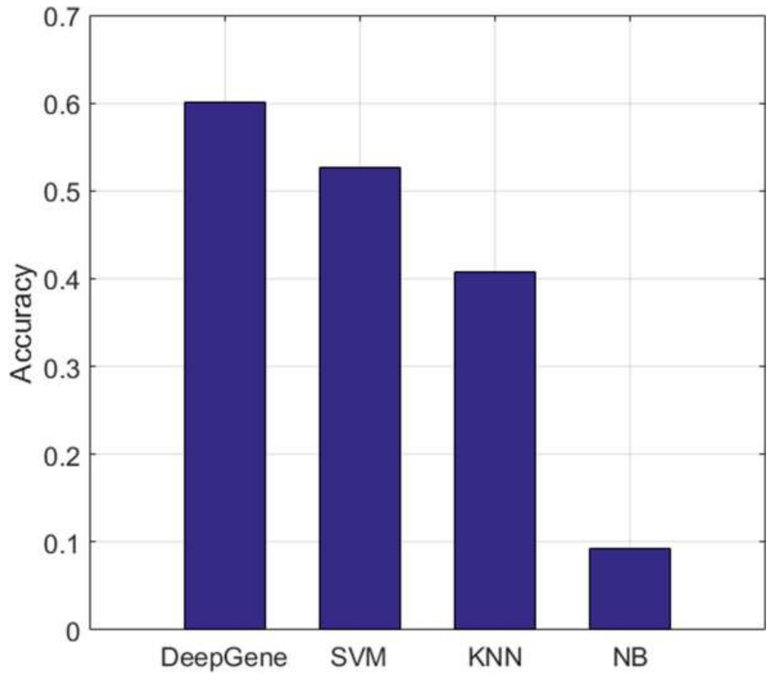

Fig. 7 Testing accuracy of DeepGene against three widely adopted classifiers with raw gene input data. All methods use raw gene data as input. The DNN classifier is still favorable against the other methods

more significant than what the CGF contributes. We attribute ISR's advantage to its remarkable reduction of the gene data sparsity. It is also mentionable that ISR exhibits more strength when combined with CGF, as the first bar in Fig. 5 indicates. This can be explained by the synergy effect of binary gene data and indexed gene data.

Furthermore, we note that ISR conducts lossless conversion when $n_{NZ} \le n_{ISR}$, i.e. the indexed data can be readily converted back to the original binary data if necessary.

Data optimization by CGF and ISR

Besides aiding our DeepGene method, the CGF and ISR steps can also benefit other classification methods for input data optimization. To evaluate the optimization effect, we apply CGF + ISR to the three classifiers SVM, KNN and NB discussed in the "Comparison with widely adopted models" Section, and record their testing accuracies before and after the input data optimization. For fair comparison, the parameters of the classifiers remain the same.

Figure 8 shows the accuracy change before and after the input data optimization of CGF + ISR. It is observed that applying CGF + ISR can notably refine the input data, thus improve the testing accuracies of the classifiers. We also note that by applying CGF + ISR, the accuracy improvements of the three classifiers are not as large as that of DeepGene. Since DeepGene is based on DNN, it is more advantageous in processing complicated data structures, thus can benefit from CGF + ISR more.

DNN classifier

The DNN classifier is the mainstay of DeepGene, which conducts the classification and generates the final output. Figure 6 has shown the significant advantage of DeepGene against three widely adopted classifiers, among which DeepGene exhibits at least 24% of performance improvement. To examine the performance of the DNN classifier itself without the pre-processing steps of CGF and ISR, we also record the accuracy of the DNN classifier with raw gene data in Fig. 7, which has shown that the DNN classifier still generates the best accuracy (60.1% against the second best 52.7% of SVM).

To further validate that the 10-fold validation accuracy of DNN is indeed higher than that of SVM, we assume that these two classifiers are independent of each other, and conduct *t*-test with the null hypothesis that these two classifiers have equal validation accuracy under the significance of 0.001. The sample standard deviation of DNN and SVM are recorded as $s_{X_1} = 1.51\% = 0.0151$ and $s_{X_2} = 2.12\% = 0.0212$, respectively. The t statistic is then calculated as:

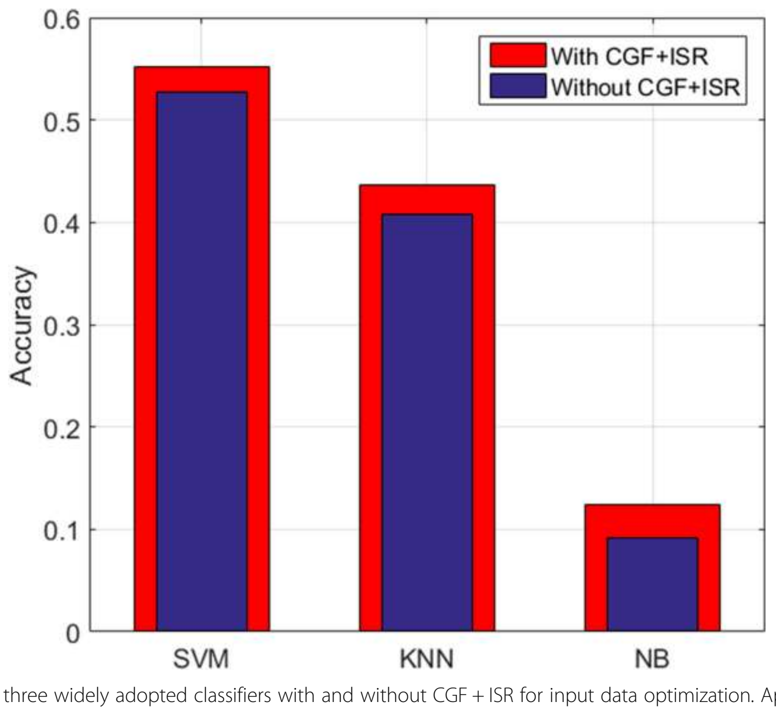

Fig. 8 Testing accuracies of three widely adopted classifiers with and without CGF + ISR for input data optimization. Applying CGF + ISR can notably refine the input data, thus improve the testing accuracies of the classifiers

$$t = \frac{\overline{X}_1 - \overline{X}_2}{s_{X_1X_2} \cdot \sqrt{\frac{2}{n}}} = \frac{0.601 - 0.527}{3.387e{-}4 \times \sqrt{\frac{2}{10}}} = 488.5,$$

where

$$s_{X_1X_2} = \frac{s_{X_1}^2 + s_{X_2}^2}{2} = 3.387e{-}4.$$

Here, the degree of freedom is $n - 1 = 9$. By checking the one-tailed significance table, the corresponding t statistic of the p-value 0.001 is 3.922, which is far less than our $t = 488.5$. Hence the null hypothesis is rejected in favor of the alternative hypothesis, and we prove that the 10-fold cross validation accuracy of DNN is indeed higher than that of SVM. It is notable that using the DNN alone is the lowest configuration of DeepGene (see Fig. 5), and SVM has the highest performance out of the three comparison classifiers. As a result, our t-test above has also proved that DeepGene is indeed higher in performance against all of the three comparison methods.

We attribute the advantage of the DNN classifier to its capacity in extracting the complex features of the input data. The multiple nonlinear processing layers make the DNN especially suitable in processing complex data that are too tough for conventional linear classifiers such as linear kernel SVM. We also note that DeepGene is just one of our initial trials for DNN-based gene data processing, but has already shown promising results against widely adopted methods. The DNN classifier has the potential to show greater advantages towards more complex (e.g. images or multi-dimensional gene data) and large-scale data to conventional classifiers, which will be discussed in our future works.

Limitation and future study

Currently DeepGene is only tested on datasets of somatic point mutations with known cancer types, i.e., the histological biopsy sites are already known. Therefore, in this study, DeepGene only demonstrates the power of capturing complex association between somatic point mutation and cancer types, and more of its application potentials will be evidenced by tumor samples with completely unknown cancer type information (such as CTC or ctDNA data) in our future works. The association between point mutation and other genetic aberrations such as copy number variance, translocation, and small insertion and deletion will also be covered in our future works. It will be proved that to a large extent, adopting point mutation alone is good enough for cancer type or subtype classification.

Conclusions

In this paper, we propose the DeepGene method for somatic point mutation based cancer type classification. DeepGene consists of three major steps. The CGF step concentrates the gene data with mutation occurrence frequency; the ISR step reduces the gene data sparsity with the indexes of non-zero elements; and in the final step, the DNN-based classifier takes the processed data and generates the classification result with high-level data feature learning.

We conduct experiments on the compiled TCGA-DeepGene dataset, which is a reformulated subset of the TCGA dataset with mutations on 12 types of cancer. Controlled variable experiments indicate that CGF, ISR and DNN classifier all have significant contribution in improving the classification accuracy. We then compare DeepGene with three widely adopted data classifiers, the results of which exhibit the remarkable advantages of DeepGene, which has achieved > 24% of performance improvement in terms of testing accuracy against the comparison classification methods.

We demonstrated the advantages and potentials of the DeepGene model for somatic point mutation based gene data processing, and we suggest that the model can be extended and transferred to other complex genotype-phenotype association studies, which we believe will benefit many related areas. As for future studies, we will refine our model for other complex and large-scale data, as well as broadening our training dataset, so that the classification result can be further improved.

Acknowledgements
We would like to thank the reviewers for their valuable suggestions and remarks, which have contributed to the improvement of our paper.

Declaration
This article has been published as part of BMC Bioinformatics Volume 17 Supplement 17, 2016: Proceedings of the 27th International Conference on Genome Informatics: bioinformatics. The full contents of the supplement are available online at http://bmcbioinformatics.biomedcentral.com/articles/supplements/volume-17-supplement-17.

Funding
This work is supported by the University of Sydney & Shanghai Jiaotong University Joint Research Alliance (SJTU-USYD Translate Medicine Fund – Systems Biomedicine AF6260003/04), and Australian Research Council (ARC) funding. This work is also supported by the National Natural Science Fund of China (NSFC 81502423 and NSFC 81272271), the China National Key Projects for Infectious Disease (2012ZX10002012-008 and 2013ZX10002010-006), the Shanghai Pujiang Talents Fund (15PJ1404100), the Chinese Education Minister-Returned Oversea Talent Initiative Fund (15001643), and the Shanghai Board of Education-Science Innovation (15ZZ014). Publication costs were funded by the National Natural Science Fund of China (NSFC 81502423).

Availability of data and material
The MATLAB source code of DeepGene and the TCGA-DeepGene dataset are both available at our website: https://github.com/yuanyc06/deepgene.

Authors' contributions
YS provided the initial ideas and fundamental materials. YY designed the algorithm, wrote the code, and conducted the experiments. YY and YS wrote the main manuscript text. CL, JK, WC, ZH and DDF revised the manuscript. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Consent for publication
Not applicable.

Ethics approval and consent to participate
Not applicable.

Author details
[1]School of Information Technologies, The University of Sydney, Darlington, NSW 2008, Australia. [2]Key Laboratory of Systems Biomedicine, Shanghai Center for Systems Biomedicine, Shanghai Jiaotong University, Shanghai 200240, China.

Published: 23 December 2016

References

1. Feuerstein M. Defining cancer survivorship. J Cancer Surviv. 2007;1(1):5–7.
2. Stewart B, Wild CP. World cancer report 2014. 2015. World.
3. DeFrancesco L. Life Technologies promises [dollar] 1,000 genome. Nat Biotechnol. 2012;30(2):126.
4. Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh ML, Downing JR, Caligiuri MA. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science. 1999;286(5439):531–7.
5. Greenman C, Stephens P, Smith R, Dalgliesh GL, Hunter C, Bignell G, Davies H, Teague J, Butler A, Stevens C. Patterns of somatic mutation in human cancer genomes. Nature. 2007;446(7132):153–8.
6. Wang Q, Jia P, Li F, Chen H, Ji H, Hucks D, Dahlman KB, Pao W, Zhao Z. Detecting somatic point mutations in cancer genome sequencing data: a comparison of mutation callers. Genome Med. 2013;5(10):91.
7. Longo DL. Tumor heterogeneity and personalized medicine. N Engl J Med. 2012;366(10):956–7.
8. Sledge GW. What is targeted therapy? J Clin Oncol. 2005;23(8):1614–5.
9. Gudeman J, Jozwiakowski M, Chollet J, Randell M. Potential risks of pharmacy compounding. Drugs R D. 2013;13(1):1–8.
10. Franken B, de Groot MR, Mastboom WJ, Vermes I, van der Palen J, Tibbe AG, Terstappen LW. Circulating tumor cells, disease recurrence and survival in newly diagnosed breast cancer. Breast Cancer Res. 2012;14(5):1–8.
11. Sleijfer S, Gratama J-W, Sieuwerts AM, Kraan J, Martens JW, Foekens JA. Circulating tumour cell detection on its way to routine diagnostic implementation? Eur J Cancer. 2007;43(18):2645–50.
12. Hayes DF, Smerage J. Is there a role for circulating tumor cells in the management of breast cancer? Clin Cancer Res. 2008;14(12):3646–50.
13. Forbes SA, Beare D, Gunasekaran P, Leung K, Bindal N, Boutselakis H, Ding M, Bamford S, Cole C, Ward S. COSMIC: exploring the world's knowledge of somatic mutations in human cancer. Nucleic Acids Res. 2015;43(D1):D805–11.
14. Watson IR, Takahashi K, Futreal PA, Chin L. Emerging patterns of somatic mutations in cancer. Nat Rev Genet. 2013;14(10):703–18.
15. Koboldt DC, Zhang Q, Larson DE, Shen D, McLellan MD, Lin L, Miller CA, Mardis ER, Ding L, Wilson RK. VarScan 2: somatic mutation and copy number alteration discovery in cancer by exome sequencing. Genome Res. 2012;22(3):568–76.
16. Browne RP, McNicholas PD, Sparling MD. Model-based learning using a mixture of mixtures of gaussian and uniform distributions. IEEE Trans Pattern Anal Mach Intell. 2012;34(4):814–7.
17. Chicco D, Sadowski P, Baldi P. Deep autoencoder neural networks for gene ontology annotation predictions. Proc ACM Conf Bioinformatics, Computational Biology. Newport Beach: Health Informatics. 2014:533–540.
18. Chow CK, Zhu H, Lacy J, Lingen MW, Kuo WP, Chan K. A cooperative feature gene extraction algorithm that combines classification and clustering. Washington, DC: IEEE Intl Conf Bioinformatics Biomedicine Workshop (BIBMW). 2009:197–202.
19. Huang Z, Huang D, Ni S, Peng Z, Sheng W, Du X. Plasma microRNAs are promising novel biomarkers for early detection of colorectal cancer. Int J Cancer. 2010;127(1):118–26.
20. Aaroe J, Lindahl T, Dumeaux V, Saebo S, Tobin D, Hagen N, Skaane P, Lonneborg A, Sharma P, Borresen-Dale A-L. Gene expression profiling of peripheral blood cells for early detection of breast cancer. Breast Cancer Res. 2010;12(1):R7.
21. Kurman RJ, Visvanathan K, Roden R, Wu T, Shih I-M. Early detection and treatment of ovarian cancer: shifting from early stage to minimal volume of disease based on a new model of carcinogenesis. Am J Obstet Gynecol. 2008;198(4):351–6.
22. Balss J, Meyer J, Mueller W, Korshunov A, Hartmann C, von Deimling A. Analysis of the IDH1 codon 132 mutation in brain tumors. Acta Neuropathol. 2008;116(6):597–602.
23. Winnepenninckx V, Lazar V, Michiels S, Dessen P, Stas M, Alonso SR, Avril M-F, Romero PLO, Robert T, Balacescu O. Gene expression profiling of primary cutaneous melanoma and clinical outcome. J Natl Cancer Inst. 2006;98(7):472–82.
24. Cho J-H, Lee D, Park JH, Lee I-B. New gene selection method for classification of cancer subtypes considering within-class variation. FEBS Lett. 2003;551(1–3):3–7.
25. Yang K, Cai Z, Li J, Lin G. A stable gene selection in microarray data analysis. BMC Bioinformatics. 2006;7(1):228.
26. Cai Z, Xu L, Shi Y, Salavatipour MR, Goebel R, Lin G. Using gene clustering to identify discriminatory genes with higher classification accuracy. Arlington: IEEE Symp Bioinformatics BioEngineering (BIBE). 2006:235–242.
27. Tao Y, Sam L, Li J, Friedman C, Lussier YA. Information theory applied to the sparse gene ontology annotation network to predict novel gene function. Bioinformatics. 2007;23(13):i529–38.
28. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97.
29. Harrow J, Frankish A, Gonzalez JM, Tapanari E, Diekhans M, Kokocinski F, Aken BL, Barrell D, Zadissa A, Searle S. GENCODE: the reference human genome annotation for The ENCODE Project. Genome Res. 2012;22(9):1760–74.
30. Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science. 2006;313(5786):504–7.
31. Deng L, Yu D. Deep learning: methods and applications. Foundations Trends Signal Processing. 2014;7(3–4):197–387.
32. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1989;1(4):541–51.
33. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. 2012. p. 1097–105.
34. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. 2014. arXiv preprint arXiv:14094842.
35. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. Columbus: IEEE Conf Comput Vision Pattern Recognition (CVPR). 2014:580–587.
36. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. 2014. arXiv preprint arXiv:14114038.
37. Sun Y, Wang X, Tang X. Deep convolutional network cascade for facial point detection. Portland: IEEE Conf Comput Vision Pattern Recognition (CVPR). 2013:3476–3483.
38. Sun Y, Wang X, Tang X. Deep learning face representation from predicting 10,000 classes. Columbus: IEEE Conf Comput Vision Pattern Recognition (CVPR). 2014:1891–1898.
39. Tomczak K, Czerwińska P, Wiznerowicz M. The Cancer Genome Atlas (TCGA): an immeasurable source of knowledge. Contemp Oncol. 2015;19(1A):A68. Last downloaded on April 8th, 2015.
40. Nair V, Hinton GE. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10). 2010. p. 807–14.
41. Bishop CM. Pattern recognition and machine learning, vol. 4. New York: Springer; 2006.
42. Vedaldi A, Lenc K. MatConvNet-convolutional neural networks for MATLAB. 2014. arXiv preprint arXiv:14124564.
43. Mostajabi M, Yadollahpour P, Shakhnarovich G. Feedforward semantic segmentation with zoom-out features. 2014. arXiv preprint arXiv:14120774.
44. Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. Am Stat. 1992;46(3):175–85.
45. Rennie JD, Shih L, Teevan J, Karger DR. Tackling the poor assumptions of naive bayes text classifiers. Washington: ICML; 2003. p. 616–23.
46. Chang C-C, Lin C-J. LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol (TIST). 2011;2(3):27.
47. Dudoit S, Fridlyand J, Speed TP. Comparison of discrimination methods for the classification of tumors using gene expression data. J Am Stat Assoc. 2002;97(457):77–87.