Deep learning for predicting disease status using genomic data

Qianfan Wu [1], **Adel Boueiz** [2,3], **Alican Bozkurt** [4], **Arya Masoomi** [4], **Allan Wang** [5], **Dawn L DeMeo** [2], **Scott T Weiss** [2], **Weiliang Qiu** [Corresp. 2]

[1] Questrom School of Business, Boston University, Boston, USA
[2] Brigham and Women's Hospital/Harvard Medical School, Boston, USA
[3] Pulmonary and Critical Care Division, Brigham and Women's Hospital/Harvard Medical School, Boston, USA
[4] Department of Computer Science, Northeastern University, Boston, USA
[5] Belmont High School, Boston, USA

Corresponding Author: Weiliang Qiu
Email address: stwxq@channing.harvard.edu

Predicting disease status for a complex human disease using genomic data is an important, yet challenging, step in personalized medicine. Among many challenges, the so-called curse of dimensionality problem results in unsatisfied performances of many state-of-art machine learning algorithms. A major recent advance in machine learning is the rapid development of deep learning algorithms that can efficiently extract meaningful features from high-dimensional and complex datasets through a stacked and hierarchical learning process. Deep learning has shown breakthrough performance in several areas including image recognition, natural language processing, and speech recognition. However, the performance of deep learning in predicting disease status using genomic datasets is still not well studied. In this article, we performed a review on the four relevant articles that we found through our thorough literature review. All four articles used auto-encoders to project high-dimensional genomic data to a low dimensional space and then applied the state-of-the-art machine learning algorithms to predict disease status based on the low-dimensional representations. This deep learning approach outperformed existing prediction approaches, such as prediction based on probe-wise screening and prediction based on principal component analysis. The limitations of the current deep learning approach and possible improvements were also discussed.

Deep learning for predicting disease status using genomic data

Qianfan Wu[1], Adel Boueiz[2,3], Alican Bozkurt[4], Arya Masoomi[4], Allan Wang[5], Dawn L. DeMeo[2], Scott T. Weiss[2], Weiliang Qiu[2*]

[1] Questrom School of Business, Boston University, 595 Commonwealth Avenue, Boston, MA, 02215

[2] Channing Division of Network Medicine, Brigham and Women's Hospital/Harvard Medical School, 181 Longwood Avenue, Boston MA 02115

[3] Pulmonary and Critical Care Division, Department of Medicine, Brigham and Women's Hospital, Harvard Medical School, Boston, MA

[4] Department of Computer Science, Northeastern University, Boston, MA

[5] Belmont High School, Boston, MA

*: Corresponding author's email address: stwxq@channing.harvard.edu

Abstract

Predicting disease status for a complex human disease using genomic data is an important, yet challenging, step in personalized medicine. Among many challenges, the so-called curse of dimensionality problem results in unsatisfied performances of many state-of-art machine learning algorithms. A major recent advance in machine learning is the rapid development of deep learning algorithms that can efficiently extract meaningful features from high-dimensional and complex datasets through a stacked and hierarchical learning process. Deep learning has shown breakthrough performance in several areas including image recognition, natural language processing, and speech recognition. However, the performance of deep learning in predicting

disease status using genomic datasets is still not well studied. In this article, we performed a review on the four relevant articles that we found through our thorough literature review. All four articles used auto-encoders to project high-dimensional genomic data to a low dimensional space and then applied the state-of-the-art machine learning algorithms to predict disease status based on the low-dimensional representations. This deep learning approach outperformed existing prediction approaches, such as prediction based on probe-wise screening and prediction based on principal component analysis. The limitations of the current deep learning approach and possible improvements were also discussed.

1. Introduction

Complex human diseases, such as cancers, cardiovascular diseases, and respiratory diseases, have caused huge public health concerns and economic burdens [1, 2]. It is believed that both environmental factors (e.g., smoking exposure, nutrient intake, physical exercise, etc.) and genomic factors contribute to the development of complex human diseases[3]. We refer genomic factors to any molecular factors related to genes, such as genotype, gene expression, DNA methylation, microRNA expression, metabolites, proteins, etc. Cutting-edge technologies, e.g., genotyping and next-generation whole genome sequencing, greatly facilitate the investigations of the associations of genomic factors to complex human diseases so that researchers can unbiasedly detect disease-associated factors. In addition to uncovering the underlying molecular mechanisms, researchers expect that the disease-associated genomic factors could also help diagnose disease, personalize treatment, and develop new medicines[4].

Several machine learning methods, such as support vector machine[5] (SVM), Random Forest[6], and K-Nearest Neighbors[7] have been successfully applied in disease prediction based on clinical data[8-10]. For genomic data generated by high-throughput technologies (Figure 1), the major challenge in disease prediction is the "curse of dimensionality"[11-13] (i.e., the number of genomic factors is far larger than the number of samples), resulting in model over-fitting and computational inefficiency.

A reasonable approach[14, 15] to handle the curse of dimensionality is to first apply feature selection techniques to select key features relevant to the disease of interest, and then to predict the disease status based on these key features (Figure 2). In genomic data analysis, a feature can be a gene probe/transcript or a (non)linear combination of several gene probes/transcripts. Traditional feature selection techniques (e.g., forward variable selection, backward variable deletion, stepwise variable selection, probe-wise tests, or principal component analysis) have limited performance in genomic data analysis. Forward variable selection, backward variable deletion, and stepwise variable selection are time-consuming. Hence they are not suitable for whole genome-wide analysis. Probe-wise tests ignore the fact that many omics variables are correlated and therefore carry redundant information regarding prediction. Ignoring the redundancy would result in the selected probes are non-reproducible in independent cohorts [13, 16, 17]. In addition, contributions of different genomic risk factors might be different, however, probe-wise tests implicitly assign equal weights to all selected probes. Principal component analysis (PCA) explicitly assigns different weights to different probes. However, PCA produces linear combination of probes and ignores the possible non-linear relationship between probes.

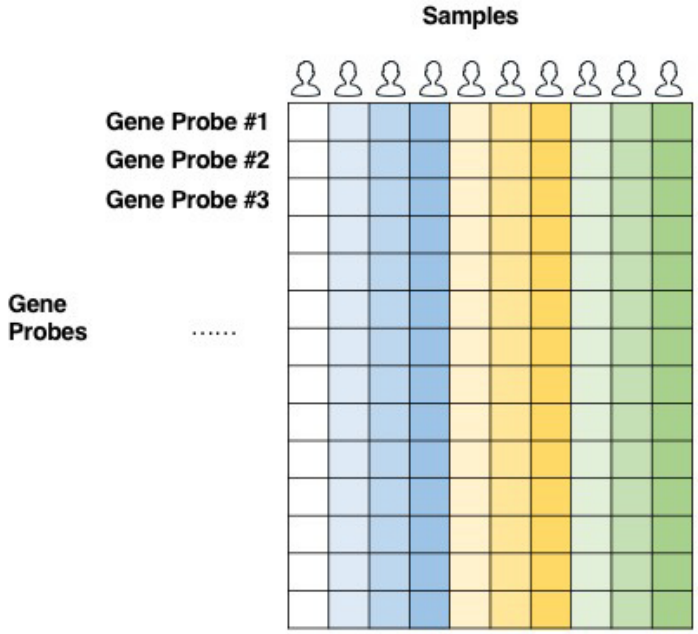

Figure 1: An illustration of gene expression data. In the above figure, each row represents 1 gene probe and each column represents one sample (one person). The (i,j) cell records the expression level of the i-th gene probe for the j-th sample. Gene expression data typically have high dimensionality (20,000 – 50,000 gene probes) and small sample size (<1000), resulting in the "curse of dimensionality problem."

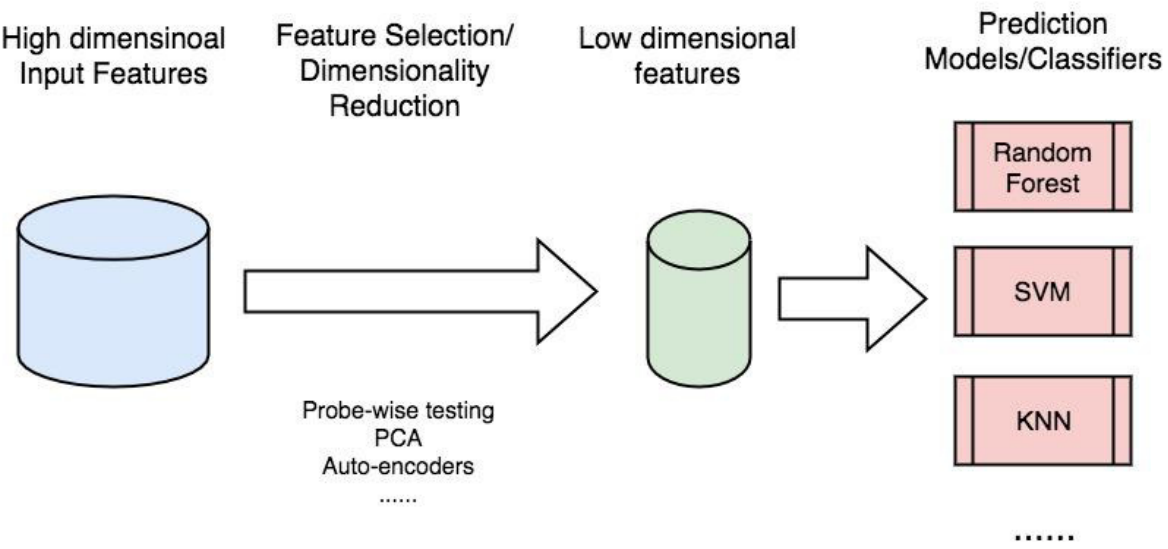

Figure 2: An illustration of building prediction models using genomic datasets. The idea is to first reduce the dimensionality of the input features and then feed the low dimensionality features into prediction model/classifiers. Dimensionality reduction techniques typically include probe-wise testing, principal component analysis (PCA), and auto-encoders.

Recently, deep learning methods have made breakthrough progress in image/video recognition[18], natural language processing[19], and robotics[20, 21]. Through a stacked and hierarchical learning system, deep learning methods could efficiently capture complex relationships between high-dimensional features, either spatial or consequential[22].

In bioinformatics, deep learning methods have fruitful and innovative applications in medical image classification[23, 24], predicting DNA- and RNA-binding proteins sequences[25], and DNA sequence noncoding variants effects predicting[26]. However, using deep learning methods to predict disease status is not a well-researched area.

Most investigators in genomic data analysis fields might hear about deep learning and would like to learn more about deep learning and how deep learning could be used to predict disease status based on genomic data. In this review, we will first introduce the main components of deep learning and the most frequently used deep learning feature extraction methods in genomic data analysis. We will then review the papers that used deep learning to predict complex human diseases based on genomic data. The limitations of the current deep learning approach and possible improvements will also be discussed.

2. Survey Methodology

To thoroughly search recent literatures on deep learning applications in disease prediction, we carefully reviewed previous works, searched popular sites: Google Scholar, PubMed, IEEE Xplore, and PMC, and examined related online blogs and tutorials, such as GitHub (http://github.com/), Kaggle (http://www.kaggle.com/), and Cross Validated (https://stats.stackexchange.com/). We identified four papers[13, 31, 46, 47] published between January 2013 and December 2017, which applied deep learning methods in disease prediction using genomic data.

Before we review the details of the four studies, we first introduce in the following sections the main components of deep learning and the most frequently used deep learning feature extraction methods in genomic data analysis.

3. Artificial Neural Networks (ANNs) and Deep Learning Methods in Predicting Disease

The main component of all deep learning algorithms is Artificial Neural Networks (ANNs). Understanding how ANNs are constructed and trained is the first step to understand deep learning methods.

Artificial Neural Networks (ANNs):

Artificial Neural Networks are computing systems that are inspired by the biological neural networks constituting brains. Typically, an ANN is a network of nodes with multilayers (one input layer, one output layer, and several hidden internal layers). Within a layer, nodes are not connected, while between the layers nodes are fully connected (*Figure 3, Figure 4*). Each node can store a value (e.g., Z_i for the i-th node in a given layer) and each edge can have a weight (e.g., w_{ji} for the edge connecting node i in the given layer with node j in the previous layer). The value of a node on a given layer, except for the first layer (i.e., the input layer), is a function of a bias (i.e., threshold; e.g., b_i for the i-th node) and the weighted average values of all nodes on the previous layer. The function is called activation function. For instance, $\hat{Y_1} = 1$ *if* $(b_i + w_{1i} * Z_1 + ... + w_{ni} * Z_n) > 0$ and $\hat{Y_1} = 0$ otherwise, where n is the number of nodes in the previous layer and Z_j is the value for the j-th node in the previous layer. Usually, activation functions, such as Sigmoid, Rectified Linear Unit (ReLU)[27], and Hyperbolic Tangent (Tanh), are non-linear.

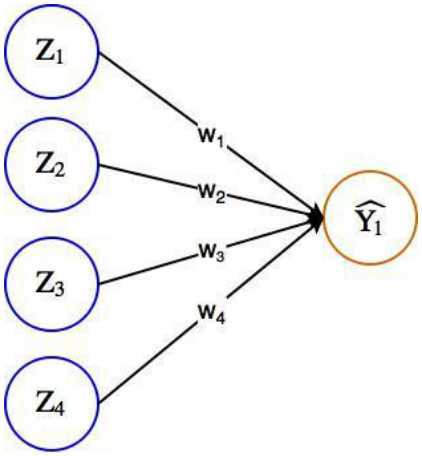

Figure 3: An illustration of a simple ANN: This simple feed-forward ANN has four input nodes and one output node. On the edges, $w_1 - w_4$ represent the weights of the input nodes. The value Y_1 for the output node is computed as $\hat{Y}_1 = f(b + Z_1 * w_1 + Z_2 * w_2 + Z_3 * w_3 + Z_4 * w_4)$, where b is the bias term, and f is the activation function.

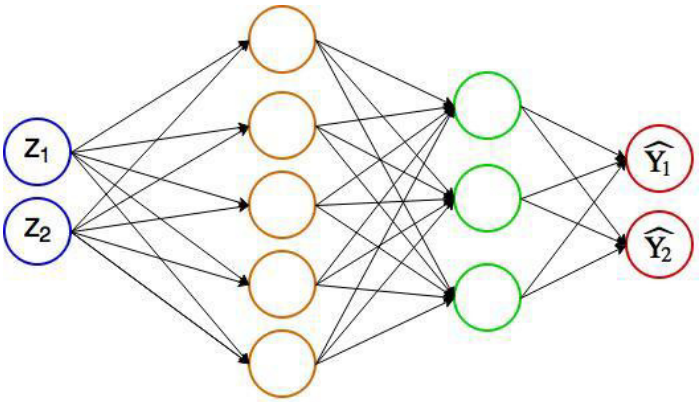

Figure 4: An illustration of a multiple-layer ANN. This multiple layer ANN has one input layer, two hidden layers, and one output layer, with each layer connected to the previous layer. The activation function f is applied to each node on the hidden layer and the output layer.

Training ANNs:

A training data set and a validation set, in which the values of the nodes in the output layers are known (e.g., 1 for a positive outcome and 0 for a negative outcome), are needed to estimate the optimal values of the biases and edge weights (i.e., to train the ANN). The idea is to find a set of biases and edge weights (parameters) that minimize the difference between the true values and predicted values of nodes in the output layer. The difference is a function of the biases and edge weights and is usually called loss function.

Gradient descent is an optimization method for updating the parameters of a neural network to minimize the loss function (Figure 5). It uses the fact that optimal parameters are achieved when gradient of the loss function with respect to the parameters are zero. However, finding parameters that are the solution to zero gradient equation is a nontrivial task for complex networks with large number of parameters. An alternative method to solving the gradient equation is, starting with an initial point, to iteratively update each parameter proportional to the negative of the gradient of the loss function with respect to the parameter, and continue this procedure until amount of change of parameters are below a predefined threshold. An important part of this method is to calculate the gradient of loss function with respect to every parameter in the network. Backpropogation is an algorithm for efficiently calculating the gradient for each parameter, using chain rule: For the simple network in Figure 3, $\frac{\partial Loss(w)}{\partial w_1} = \frac{\partial Loss(w)}{\partial \hat{Y}_1} \frac{\partial \hat{Y}_1}{\partial w_1}$, where $Loss(w)$ is the loss function. This implies that once we know the gradients at some layer, we can easily calculate the gradients for the layer before it.

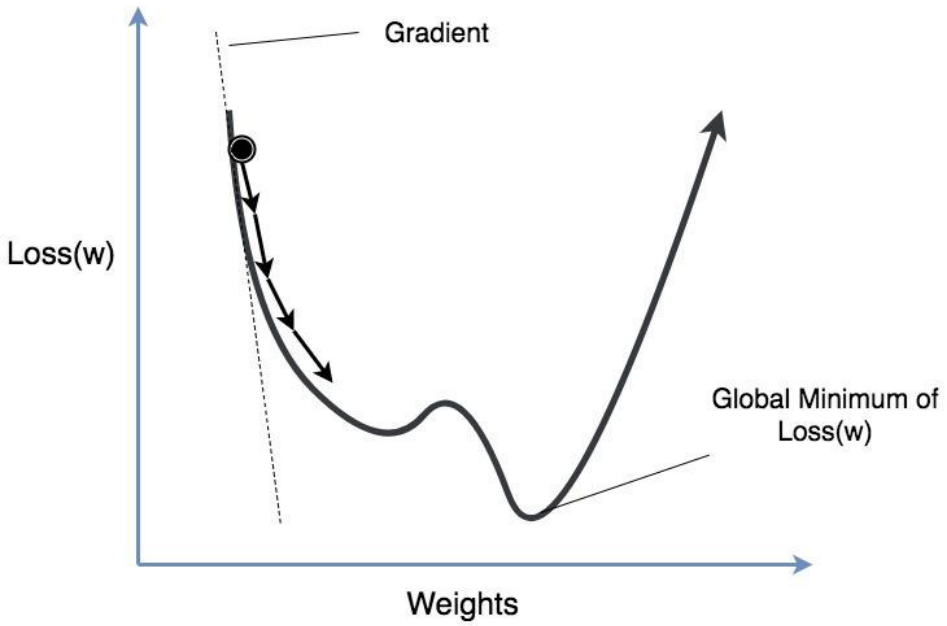

Figure 5: Gradient Descent Training. The x-axis is the weight w and the y-axis is the loss function Loss(w). In Gradient Descent optimization, learning rate represents how much the edge weights are adjusted in each step before the global minimum is achieved. Learning rate could also be seen as the "step size" in the learning process. With a higher learning rate, the gradients are adjusted by a greater amount each step. With a lower learning rate, the gradients are adjusted by a smaller amount each step.

Deep Learning and Deep Neural Networks (DNNs):

ANNs with only one or two hidden layers have a shallow architecture, which contains only two levels of data-dependent computational elements and can be very inefficient regarding the number of computational units (e.g., hidden nodes), and in terms of required training examples[11]. In contrast, ANNs with more than two hidden layers (i.e., deep neural networks) have a deep architecture, which can compactly represent a large number of computational elements via the composition of many nonlinearities[11]. Deep learning methods are defined as computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction[22].

The performance of deep learning relies on the methods to train the parametersin DNNs. Intuitively, we can train the parameters by minimizing the prediction error rates (the loss function) through applying gradient descent. However, empirical experiments showed that this supervised approach has poor performance for DNNs[11, 28], in the regime where number of input features are comparable to (or even far larger than) number of training samples, which is the case in genomic datasets. In contrast, unsupervised learning at each stage of a deep network proposed by the seminal works of Hinton et al. (2006)[29] and Hinton and Salakhutdinov (2006)[30]pretrains each hidden layer as the encoder of an auto-encoder trying to reconstruct the output of the previous layer. . Hence, combining unsupervised approach with the supervised approach, such as combining an auto-encoder with a supervised fine-tuning phase (i.e., fine-tune all the parameters of the ANN using backpropagation and gradient descent on a global supervised cost function), can significantly improve the performance of deep learning methods for data-sparse datasets[11, 28].

Auto-encoder (AE):

An auto-encoder is a type of ANN that aims to find a new representation of input nodes (e.g., gene probes in genomic data analysis) in an *unsupervised* manner, from which the input can be reconstructed without too much loss of information[28]. Like ANN, an auto-encoder has one input layer, one output layer, and one or multiple hidden layers (*Figure 6*). Suppose $\boldsymbol{X}$ is the original data in a p-dimensional space. An auto-encoder would first project $\boldsymbol{X}$ to a q-dimensional space $\boldsymbol{Y}=g_1(\boldsymbol{X})$, where g_1 is a non-linear projection function. Then it transforms $\boldsymbol{Y}$ back to the p-dimensional space $\boldsymbol{Z}=g_2(\boldsymbol{Y})$, where g_2 is also a non-linear projection function. The optimal projection $\boldsymbol{Y}^*$ minimizes the loss function $loss[\boldsymbol{X}, g_2(\boldsymbol{Y})]$ that measures the differences between $\boldsymbol{X}$ and $\boldsymbol{Z}=g_2(\boldsymbol{Y})$. Note that since q is different from p, both the projection function g_1 and the projection function g_2 are not one-to-one mapping functions. Hence, the inverse functions g_1^{-1} and g_2^{-1} do not exist.

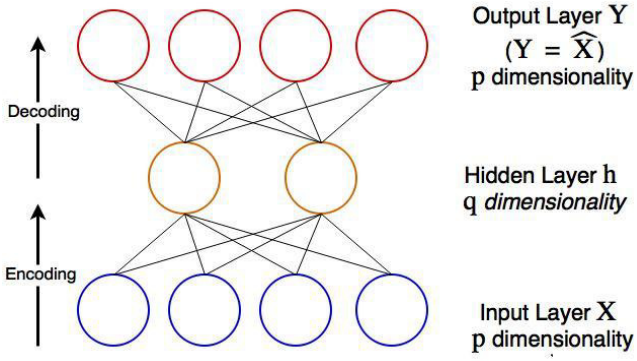

Figure 6: Illustration of a basic auto-encoder. This auto-encoder has 2 hidden units. $\boldsymbol{X}$ *is the inputs,* $\boldsymbol{Y}=\hat{X}$ *is the reconstructed inputs in the output layer, h is the hidden layer. The dimension of the original input data is reduced from p=4 to q=2. The optimal representation in the q-dimensional space is obtained by minimizing the difference between the inputs* $\boldsymbol{X}$ *and the reconstructed inputs* $\boldsymbol{Y}$.

Similar to training ANNs, backpropagation and gradient descent can be applied to train an auto-encoder, in which the output layer has the dimension as the original data $\mathbf{Z}= g_2(\mathbf{Y})= g_2(g_1(\mathbf{X}))$.

The nodes $\mathbf{Y}=g_1(\mathbf{X})$ within the hidden layer are the representations of original features. The hidden layer is "under-complete" if the number (q) of nodes in the hidden layer is smaller than that (p) in the input layer ($q<p$). In most cases, auto-encoder outperforms Principal Component Analysis in processing high dimensional complex datasets because auto-encoder performs both linear and non-linear projections, while PCA performs only linear projection. Auto-encoders have been successfully used to efficiently extract meaningful features in disease diagnosis based on high-throughput genomic data[31, 32].

Sparse auto-encoder (Sparse AE):

Performing backpropagation and gradient descent could be inefficient if there are too many free nodes with complex dependencies in each layer[33, 34]. Sparse auto-encoder is developed to restrict the number of hidden nodes to be activated by introducing sparsity-constraints on the hidden units (*Figure 7*). Sparse auto-encoder have been proved to have favorable performance in image recognition[35] and speech emotion recognition[36], due to its efficiency in extracting meaningful features from high-dimensional data.

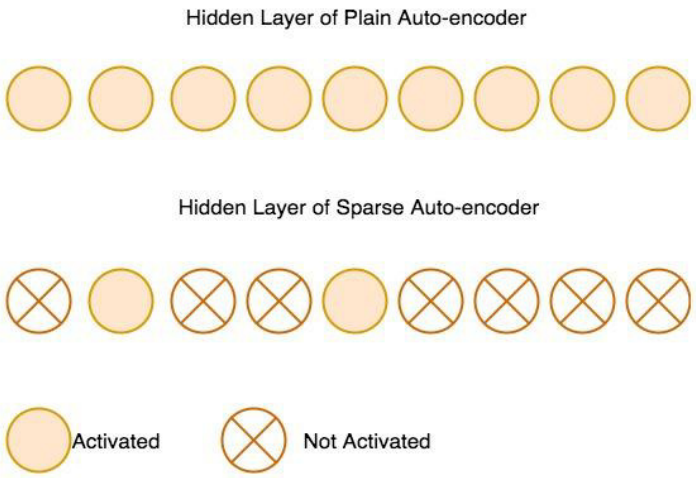

Figure 7: Illustration of a sparse auto-encoder: A sparse auto-encoder restricts the number of hidden layers activated by adding a sparsity term to the loss function. The sparsity term set the expected activation value of the hidden nodes to a small constant so that most of the hidden nodes' activations are near zero. Hence, very few hidden nodes are activated in a sparse auto-encoder.

Stacked auto-encoder (Stacked AE):

A stacked auto-encoder[11, 37, 38] is a multi-layer auto-encoder, each hidden layer of which is a representation of previous layer obtained by an auto-encoder with one hidden layer (Figure 8). The training of stacked auto-encoders is often completed by applying the greedy layer-wise pre-training approach[11]. Given extremely high-dimensional input data, a stacked auto-encoder could extract features layer by layer and finally forms a better representation to be passed into classifiers.

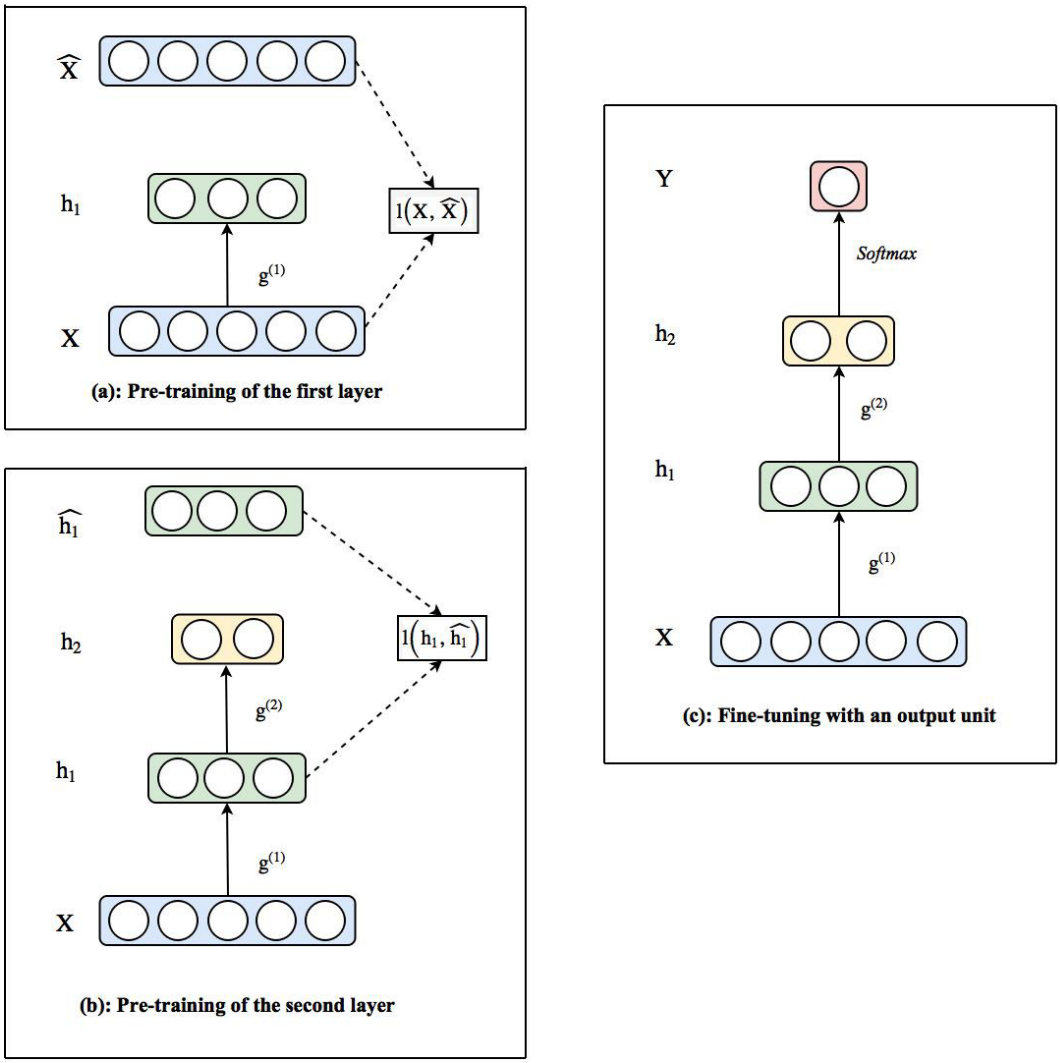

Figure 8: Illustration of stacked auto-encoder and greedy layer-wise pre-training: The stacked auto-encoder has 2 hidden layers h_1 and h_2. Under the greedy layer-wise pre-training, hidden layer h_1 is first trained under the same way as training a simple 1-layer auto-encoder by minimizing $l(X,\hat{X})$. The function $g^{(1)}$ that maps X to h_1 is learned from the first layer training, which is shown in (a). Then nodes values on h_1 are passed to the second layer h_2 to train the function $g^{(2)}$ that maps h_1 to h_2 by minimizing $l(h_1, \hat{h_1})$, which is shown in (b). After pre-training all hidden layers, an output unit Y, which serves as a classifier, could be wired on top of the hidden layers to make predictions. The whole architecture could be fine-tuned together using backpropagation and labeled data, which is shown in (c).

Denoising auto-encoder (DA):

A basic auto-encoder could successfully retain much of the information from the inputs in new features within the hidden layer. However, Vincent et al. (2010)[38] demonstrated that simply retaining information from the inputs does not guarantee that the extracted features are "good features", which could achieve high-performance in supervised learning tasks. Denoising auto-encoder has been proposed to overcome this challenge by generating a noisy representation based on the inputs, such as setting values to 0 for a small proportion of input nodes or adding a noise term with a Gaussian distribution, and then feeding the noisy term into the auto-encoder (*Figure 9*). With the introduction of the noise term to the original inputs, denoising auto-encoders construct more robust feature representations and thereby could generalize better to unseen examples and datasets.

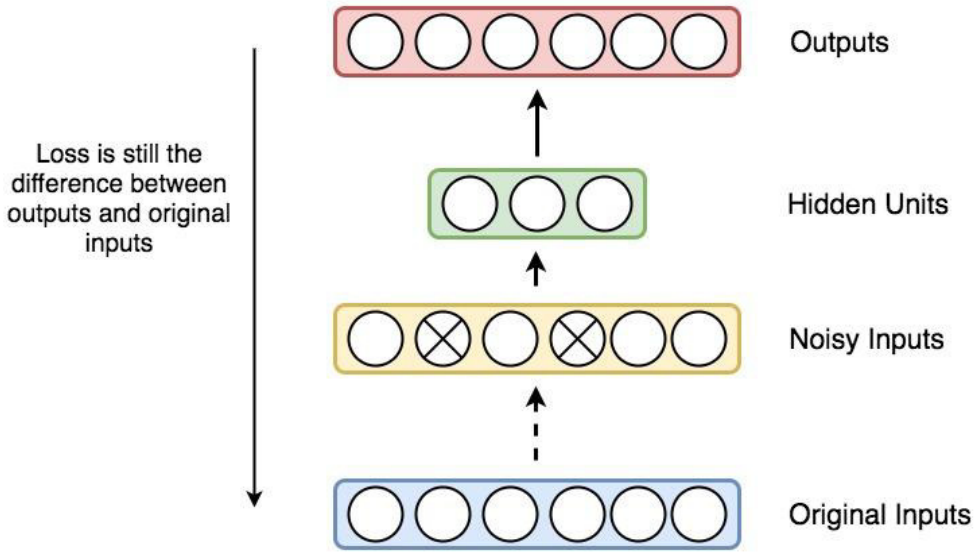

Figure 9: Illustration of a denoising auto-encoder. A denoising auto-encoder first transforms original inputs into noisy inputs. However, the loss in each step of the training process is still computed by the difference between the reconstructed representations in the output layer and the original inputs in the input layer.

Stacked denoising auto-encoder (SDAE):

A SDAE is a multi-layer auto-encoder, each hidden layer of which is a representation of the previous layer obtained by a denoising auto-encoder with one hidden layer. For example, when pre-train the 2 hidden layers h_1 and h_2 in Figure 8, one could add a noise term to the pre-training inputs X and $h_1$to construct SDAE. Vincent et al. (2010)[38] showed that the features extracted by SDAE are stable and robust under noisy inputs, by achieving the best classification results under 9 out of 10 image databases. These features could efficiently capture useful information in the input distribution and have yield equivalent or better classification performance over most of the image data processing benchmarks.

Table 1 summarizes the 5 auto-encoders described above.

Table 1. A summary of different auto-encoders

Method	Description
Regular auto-encoder (AE)	Find low-dimensional representation of input using an unsupervised approach (i.e., no outcome information is used)
Sparse AE	Restrict the number of hidden nodes to be activated to avoid too many free nodes with complex dependencies in each layer
Stacked AE	Each hidden layer is a low-dimensional representation of the previous layer obtained by AE
Denoising AE (DA)	Introduce noises to input to make AE more robust to noises
Stacked denoising AE (SDAE)	Combine stacked AE and DA (i.e., introduce noises to input in a stacked AE)

4. Deep Learning Applications in Disease Prediction

Previous Works of Disease Prediction in Genomic Data Analysis using non-deep learning approach:

Plenty of methods have been proposed in disease prediction using genomic data (e.g., [39-44]). Due to the large number of predictors (i.e., gene probes), the main approach in disease detection/prediction is to first obtain a subset of gene probes (e.g., a few top gene probes in probe-wise tests) or a subset of representations of gene probes (e.g., a few top principal components), and then to predict disease status based on the selected probes or representations using machine learning algorithms.

Furey et al. (2000)[39] used SVMs to classify cancer tissue samples using gene expression datasets. The study showed that SVMs are able to classify tissue and cell types based gene expression data and have similar performances to other machine learning methods. Khan et al. (2001)[40] was among the first to adopt basic ANNs (ANNs without hidden layers) to classify cancer samples and to identify relevant genes. In their study, the 10 top PCA components were used as inputs to the ANN to classify the small, round blue-cell tumors (SRBCT) to four distinct diagnostic categories. All 63 samples in the training set and all 25 samples in the independent testing set were correctly classified based on the 96 selected genes. Pal et al. (2007)[41] proposed to combine modified perceptron network and relational fuzzy clustering algorithms[45] to select a gene subset used for cancer subgroup classification. They applied their method to the SRBCT dataset analyzed by Khan et al. (2001)[40] and identified 7 genes that can accurately classify the samples in both training set and testing set. Chang et al. (2011)[42] used an ANN with one hidden layer coupled with an additive step-wise approach for predicting colorectal cancer (CRC) using microRNAs (miRNAs). Three miRNAs were identified with median accuracy 100% by using an extensive Monte Carlo cross-validation strategy. Sharma et al. (2012)[15] proposed a top-r feature selection technique that repeatedly divides and merge gene

expression data to select the gene subset minimizing the loss of information. The selected genes are then tested on three tumor datasets and achieved higher accuracies than other feature selection methods, such as probe-wise tests. Nanni et al. (2012)[43] examined the SVM classification performance using multiple feature reduction and data transformation approaches, including neighborhood preserving embedding, orthogonal wavelet coefficients, and texture descriptors. The study showed that a combination of different feature extraction methods could enhance genomic classification performance. For instance, the two combined methods achieved the highest average area under ROC curves (AUC = 92.18% for the WF method and 92.07% for the FUS method), while the AUC values for the 8 individual feature extraction methods were ranged from 79.24% to 91.85%. Jordan and Do (2018)[44] reviewed the studies that predict disease using full genomic information. Their review focused on polygenic risk scores (PRS), which is the most common method of integrating information from across the genome into a single estimate of genetic risk. A PRS is a weighted average of the genetic status at each associated risk locus. The weighting of each locus is usually the regression coefficient of GWAS association for the locus. Jordan and Do (2018) mentioned that the power of most PRSs to predict disease risk has been very low due to several reasons, such as small sample size, genetic ancestry, heterogeneity of risk factors and causation.

The main limitations of these previous works[13] include (1) ignoring potential non-linear relationships among the features; (2) ignoring the contribution of features with weak signals to distinguish diseases; and (3) over-simplifying the complex prediction problem, such as using single-layer ANNs.

366 **Deep Learning Applications in Disease Prediction**

367

368 Through thorough literature search, we identified four papers[13, 31, 46, 47] published between
369 January 2013 and December 2017, which applied deep learning methods in disease prediction
370 using genomic data (Table 2). The details of the four studies will be discussed below.

371

372 Fakoor et al. (2013)[13] is among the first to apply deep learning methods to extract key features
373 from gene microarray data in predicting cancers. Fakoor et al. (2013)[13] first applied PCA to
374 eliminate the effects of redundant and noisy dimensions, then applied three auto-encoders
375 methods (a sparse auto-encoder with one hidden layer, a stacked auto-encoder with 2 hidden
376 layers, and a stacked auto-encoder with fine tuning) to further extract non-linearly-correlated
377 discriminating features based on the top principal components combined with some randomly
378 selected original features, and finally used softmax regression to do classification based on the
379 low-dimensional representations. Thirteen gene microarray datasets (the range of sample sizes is
380 20 – 1,047; the range of the numbers of features is 2,000 – 54,613) were used to compare the
381 performances of deep learning methods and two traditional prediction methods: Softmax based
382 on the top principal components (PCA+Softmax) or SVM with Gaussian kernel based on the top
383 principal components (PCA+SVM). Ten-fold cross-validation was applied to estimate the
384 average and standard deviation of the prediction accuracies and compared the average ACCuracy
385 (ACC) of the three deep-learning methods with the maximum of the accuracy of the two
386 traditional methods. For 8 of the 13 genomic datasets, at least one of the three deep learning
387 methods has significantly higher average accuracy than the maximum accuracy of PCA+Softmax
388 and PCA+SVM. The median [min, max] increase of average ACC is 1.5% [0.7%, 8.3%]. The

sample sizes of the 8 datasets range from 20 to 1,047. However, stacked auto-encoder without fine-tuning usually had much worse accuracy than the traditional methods. The stacked auto-encoder with fine-tuning achieved the best accuracy in six datasets with ACC ranging from 76.67% to 95.15%, while the single-layer sparse auto-encoder perform the best in 5 datasets with ACC ranging from 46.76% to 91.50%.

Tan et al. (2015)[31] used denoising auto-encoders to learn compact and efficient representations in predicting disease status. Tan et al. (2015)[31] used the Molecular Taxonomy of Breast Cancer International Consortium (METABRIC) cohort as the training set (1,424 samples) and the testing set (712 samples) and the cohort from The Cancer Genome Atlas (TCGA) as the independent evaluation set (547 samples*)*. The DA used in Tan et al. (2015)[31] has four layers: an input layer, a corrupted input layer (i.e., noisy input layer), a hidden layer, and a reconstructed input layer. Each node in the hidden layer was used to predict disease status (e.g., tumor vs. non-tumor, or ER+ vs. ER-) depending on whether the node value for a sample in the evaluation set is greater than the optimal threshold that was obtained based on the discovery set and testing set. Tan et al. (2015)[31] showed that each of the top three hidden nodes in the discovery set could also have high prediction accuracy (> 0.9) in the evaluation set when they used their method to predict tumor status (tumor sample vs. non-tumor sample).

Danaee et al. (2016)[46] used SDAE to transform high dimensional, noisy RNA-seq gene expression data to lower dimensional, meaningful representations, based on which they applied different machine learning methods to classify breast cancer samples from the healthy control samples. They also identified a set of “Deeply Connected Genes” (DCGs) that have strongly

propagated influence on the reduced-dimension SDAE-encoding. Inspired by the classic study that applies SDAE to extract features in image data[38], Danaee et al. (2016)[46] built a SDAE model with four stacked layers of dimensions of 15,000, 10,000, 2,000, and 500, to obtain representations of genomic features to be fed into classifiers. A RNA-seq dataset (1,210 samples: 1,097 breast cancer samples and 113 healthy samples) from TCGA is used to train and validate the model in the study. Danaee et al. (2016)[46] compared their prediction method with prediction methods based on PCA, Kernel PCA (KPCA, a non-linear PCA), the 206 differentially expressed genes (DIFFEXP0.05) that were significant at an FDR of 0.05 in gene-wised tests, and top 500 most significant differentially expressed genes (DIFFEXP500). Three classifiers, including a single-layer ANN, SVM, and SVM-RBF (SVM with a radial basis function kernel), were used to do the prediction based on extracted features. Like Tan et al. (2015)[31], Danaee et al. (2016)[46] used a training set and a testing set to train classifiers, and used a validation set to evaluate the performance of the prediction methods. The classification result shows that the low-dimensional representations by SDAE outperformed other four sets of extracted features. For example, SDAE+SVM-RBF had accuracy (98.26%), sensitivity (97.61%), specificity (99.11%), precision (99.17%), and F-score [48] (0.983). Furthermore, Danaee et al. (2016)[46] showed that DCGs had slightly lower prediction accuracy than SDAE-extracted low-dimensional representations, but much higher prediction accuracy than the other methods.

Singh et al. (2016)[47] applied a stacked sparse auto-encoder (SSAE) to extract features to predict disease status for each of 36 datasets from the Gene Expression Machine Learning Repository (GEMLeR)[49]. The SSAE used by Singh et al. (2016)[47] has three hidden layers.

The input layer contains top 800 features selected based on Individual Training Error Reduction (ITER) ranking. The three hidden layers have 700, 600, and 500 nodes, respectively. The three classifiers, Softmax Regression, kernel SVM, and Random Forest, were applied to the 500 extracted features to perform binary classification. Singh et al., (2016)[47] applied 10-cross-validation to estimate the classification accuracy and area under the ROC curve (AUC). Compared with the benchmark classification results taken from the GEMLeR website[49], the deep learning approach achieved slightly higher performance: ACC > 90.8% for 35 datasets (ACC>83.7% for all 36 datasets), and AUC>90.2% for 34 datasets (AUC >79.6 for all 36 datasets).

Software packages for deep-learning-based feature extraction

Since deep learning algorithms usually are complicated, it is important to have open-source software packages available so that investigators can directly use these packages to their genomic data analysis. Both Tan et al. (2015)[31] and Danaee et al. (2016)[46] used *Theano* software that provides the implementation of auto-encoder algorithms. Fakoor et al. (2013)[13] and Singh et al. (2016)[47] did not mention the software packages that they used for auto-encoding.

Several software packages/libraries are available to build auto-encoder models and fine-tune model parameters, such as Python packages (*Scikit_learn*, *Theano*, *Keras*, and *TensorFlow)* and R packages (*h2o*, *kerasR*, and *autoencoder*). Wikipedia provides a table of deep learning software (https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software).

5. Discussion

In this article, we aimed to review all papers that applied the deep learning approach to predict disease status based on genomic data, which first obtains low-dimensional representations of high-dimensional genomic features, and then inputs these representations to the state-of-art classifiers that have excellent performance in low-dimensional classification problems. We found only 4 such papers, indicating that it is still in its infancy to predict disease status using deep learning on genomic data. However, the results of these 4 papers showed that the deep learning approach could extract useful genomic features from high-throughput whole genome data for prediction purpose with high accuracy.

Compared with commonly-used dimension-reduction methods (such as PCA and probe-wise testing), the deep learning approach could have better performance in terms of a variety of accuracy measurements: accuracy, AUC, sensitivity, specificity, precision, and F-score. Especially, it is impressive that probe-wise testing, which is currently the most popular approach to identify disease-associated probes, performed poorly compared with PCA or auto-encoders [46]. However, whether the performance of the deep learning approach is significantly better than the commonly used approaches was not investigated in the 4 papers, among which only Fakoor et al. (2013)[13] provided standard errors for the estimated ACC. However, Fakoor et al. (2013)[13] neither provided some key details (e.g., the number of principal components used and the number of randomly selected raw features), nor provided p-values for testing if the mean ACC obtained using a deep learning approach is significantly better than that by using the PCA approach. Moreover, Fakoor et al. (2013)[13] showed that not all auto-encoders could

outperform PCA. For example, Table 2 of Fakoor et al. (2013)[13] showed that for the first dataset, mean ACC (standard error) is 74.36% (0.062%) by using PCA+sparse auto-encoder, 51.35% (0.019%) by using PCA+stacked auto-encoder, while PCA approach had mean ACC 94.04% (SE 0.03%), although PCA+stacked auto-encoder with fine tuning (95.15% (0.047%)) performed better than PCA.

Different auto-encoders were used in the 4 papers, such as sparse auto-encoder, stacked auto-encoder, stacked auto-encoder with fine-tuning, denoising auto-encoder, stacked denoising auto-encoder, and stacked sparse auto-encoder. Except Fakoor et al. (2013)[13], the other three papers did not compare the auto-encoders used in the paper with other auto-encoders. Table 2 of Fakoor et al. (2013)[13] showed that PCA+stacked auto-encoder performed worse than PCA+sparse auto-encoder and PCA+stacked auto-encoder with fine-tuning in 12 of the 13 datasets. However, neither PCA+sparse auto-encoder nor PCA+stacked auto-encoder with fine-tuning could outperform each other in all 13 datasets. For a fair comparison, it could be beneficial for future studies to compare the deep learning methods mentioned above using the same datasets.

All four papers mentioned the number of hidden layers and the number of nodes in each hidden layer used for the auto-encoders. However, no justifications and guidance were given on why choosing those specific numbers of hidden layers and those specific numbers of nodes in each hidden layer. This is probably one of the main reasons why deep learning has not been widely used in the genomic research area. There are some existing methods to choose the number of layers and nodes, such as (1) starting from a small neural network and adding layers and nodes until the error stops decreasing, and (2) starting from a big neural network and remove layer and

nodes until the error increases significantly[50]. Optimization methods such as grid search and random search are also proposed and discussed[51] to optimize the parameters in model training. However, these methods are still not well studied in genomic data analysis and could not eliminate the risks of over-fitting and under-fitting. Future research is still needed in choosing and optimizing deep learning parameters, especially in genomic data analysis.

Another possible reason why deep learning has not been widely used in the genomic research area is the lack of software packages that implement deep learning algorithms for genomic data analysis. Many investigators in genomic research area use the R language and use packages in Bioconductor (a repository of R packages specifically for genomic data analysis). Although there are a couple of R packages, such as *keras* and *kerasR*, connecting R to the Keras deep learning library, there is lack of examples and tutorials on how to use them to analyze genomic data and to visualize the low-dimensional representations that are obtained by auto-encoders.

It is a non-trivial task to interpret the low-dimensional representations (features) of the original expression data obtained by auto-encoders because the representations are non-linear functions of gene probes and the hidden layers in deep learning algorithms are like “black box”[52]. Among the 4 papers that we reviewed, Tan et al. (2015)[31] and Danaee et al. (2016) [46] suggested interpreting the representations based on the probes having strongly propagated influence on the reduced-dimension auto-encoding. However, no details were given on how to select these probes, except that these probes have high edge weights.

To evaluate classification performance, several measurements were used in the four papers that we reviewed, including accuracy (ACC), area under the ROC curve (AUC), sensitivity, specificity, precision, and F-measure. When the dataset is imbalanced (i.e., number of cases/positive samples is much different from that of controls/negative samples), using ACC could be biased. For example, given a dataset with 99% true negative samples and 1% true positive samples, a classifier could achieve 99% ACC even if it wrongly classifies all the true positive samples to the negative group. Fakoor et al. (2013)[13] only used ACC as the performance metric, while several genomic datasets analyzed in Fakoor et al. (2013)[13] are imbalanced. Tan et al. (2015)[31] also only used ACC to evaluate the performances of different prediction methods, while both the training and testing datasets are highly imbalanced. For imbalanced data, other performance metrics can be used, such as AUC, F-measure, and G-measure[48, 53], which are less sensitive to the case/control imbalance.

Over-fitting is a big issue in prediction. Using the same data set to both train the prediction model and evaluate the performance of the prediction model usually causes over-estimation of the prediction accuracy. Ideally, a testing set from a population independent of the training population is required in evaluating prediction accuracy. However, genomic data are usually expensive to collect. Hence, it is usually hard to obtain independent testing set in genomic research. Thanks to the policy of the National Institute of Health of the United States, numerous genomic datasets are now publicly available in the Gene Expression Omnibus (https://www.ncbi.nlm.nih.gov/geo/), an online repository of genomic datasets. Other public genomic repositories are also available, such as TCGA (https://cancergenome.nih.gov) and GTEx (https://www.gtexportal.org/home/). Hence, nowadays it is relatively easy to obtain an

independent testing set for most of complex human diseases. Among the 4 papers that we reviewed, only Tan et al. (2015)[31] used an independent testing set. The other 3 papers used K-fold cross-validation technique to alleviate the over-fitting issue.

Genomic data usually contain technical noises, such as batch effects (large samples have to be handled in multiple batches due to capacity limits of machines). Several methods, such as ComBat[54], have been proposed to remove the effects of technical batches before downstream data analysis. We can apply ComBat to the training set and the testing set, separately. Suppose after removing technical noises we build and validate a prediction model based on the training set and the testing set, with excellent prediction accuracy. Now a new subject's genomic data are obtained. Can we apply the prediction model to this new subject? The answer probably is "no", since we do not know how to remove technical noises for only one new sample. One possible solution is to collect genomic data for a batch of subjects together. Then we can apply the prediction model to subjects in this batch after removing possible batch effects. A possiblely better solution is to improve the technology to reduce technical noises. With the advancements in sequencing technology and a rapid decline in sequencing costs, DNA sequencing has gained remarkable popularity among biomedical researchers. Compared to microarrays, DNA sequencing data is believed to deliver faster, more complete, and more scientifically accurate genomic analysis[55].

The four deep-learning papers identified in this review compared the performances of deep learning approaches with PCA approach and probe-wise test approach. There are many more

advanced feature selection methods in the literature, such as the stable feature selection method [16] and the Boruta algorithm [17]. More comprehensive comparisons are warranted.

Recently, semi-supervised learning and reinforcement learning are receiving a lot of attention in image recognition, gaming, and robotics[56-58]. How to apply the frontier deep learning innovations to genomic data analysis could be an interesting future research topic[59].

6. Conclusion.

In summary, this review showed that applying deep learning to find a low-dimensional representation for high-throughput genomic data is a promising future trend in disease prediction based on high-dimensional genomic data. The low-dimensional representation obtained by deep learning could capture both linear and non-linear relationship among the probes. Deep learning is a new technology for most scientists in genetics. Scientists in genetics should collaborate to understand how deep learning could help predict disease status using genomic data, hence to move this field forward.

Table 2. Summary of the four studies that applied deep learning to predict disease status in the genomic research

Author/Year	Datasets	Total Number of Samples	Feature Extraction Method	Classifier	Using cross-validation or not	Performance based on deep learning	Traditional Methods compared with
Fakoor:2013	gene expression data from 13 datasets	Various number of samples (20 - 1047) and	PCA+Sparse Auto-encoder PCA+Stacked	Softmax regression	10-fold cross-validation to evaluate	ACC+/- standard error: (33.7%+/-	Dimensional Reduction: PCA

		various number of features (2000 – 54675) across 13 datasets	Auto-encoder PCA+Stacked Auto-encoder with Fine-tuning	SVM with Gaussian kernel	classification performance	0.038%) – (97.5%+/- 0.079%)	Classifier: Softmax regression SVM
Tan:2015	Training: A gene expression dataset from METABRIC (Illumina HT-12 v3 platform) Independent Testing: A gene expression dataset from TCGA	METABRIC: 2136 samples (1992 breast cancer specimens and 144 tumor-adjacent normal tissues); 2520 genes after data cleaning TCGA:547 samples (522 primary tumors, 3 metastatic tumors, and 22 tumor-adjacent normal samples); 2520 genes after data cleaning	Denoising Auto-encoder (DA)	Sigmoid Activation	10-fold cross-validation to determine the appropriate parameter setting for the training set	ACC in testing set: 75.0%-99.6%	N/A

Danaee:2016	An RNA-seq expression dataset from TCGA	1210 samples (1097 breast cancer samples and 113 healthy samples)	Stacked Denoising Auto-encoder(SDAE)	ANN SVM SVM-RBF	5-fold cross-validation to evaluate classification performance	ACC: 96.95%-98.26% Sensitivity: 97.21%-98.73% Specificity: 95.29%-99.11% Precision: 95.42%-99.17% F-measure 0.970-0.983	PCA KPCA Differentially Expressed Genes
Singh: 2016	36 gene microarray datasets from GEMLeR (Affymetrix GeneChip U133 Plus 2.0 arrays)	1545 samples (9 cancers, no control samples); 54676 features	Stacked Sparse Auto-encoder(SSAE)	Softmax Regression Random Forest Linear SVM RBF SVM	10-fold cross-validation to evaluate classification performance	AUC: 80%-100% ACC: 76%-100%	KNN; SVM-RFE

References:

1. Ferlay, J., et al., *GLOBOCAN 2012 v1.0, Cancer Incidence and Mortality Worldwide: IARC CancerBase*. 2013, International Agency for Research on Cancer: Lyon, France.
2. Stewart, B. and C. Wild, *World cancer report*. 2014, International Agency for Research on Cancer: Lyon, France.
3. World Health Organization. *Genes and noncommunicable diseases*. Genes and human disease [cited 2018 April, 15]; Available from: http://www.who.int/genomics/public/geneticdiseases/en/index3.html.
4. Whitcomb, D.C., *What is personalized medicine and what should it replace?* Nat Rev Gastroenterol Hepatol, 2012. **9**(7): p. 418-24.
5. Cortes, C. and V. Vapnik, *Support-vector networks.* Machine learning, 1995. **20**(3): p. 273-297.
6. Breiman, L., *Random forests.* Machine learning, 2001. **45**(1): p. 5-32.
7. Altman, N., *An introduction to kernel and nearest-neighbor nonparametric regression.* The American Statistician, 1992. **46**(3): p. 175-185.
8. Mastoi, Q.U., et al., *Automated Diagnosis of Coronary Artery Disease: A Review and Workflow.* Cardiol Res Pract, 2018. **2018**: p. 2016282.
9. Zhang, Y.D., et al., *An imaging-based approach predicts clinical outcomes in prostate cancer through a novel support vector machine classification.* Oncotarget, 2016. **7**(47): p. 78140-78151.
10. C, G., et al., *Model-based and Model-free Machine Learning Techniques for Diagnostic Prediction and Classification of Clinical Outcomes in Parkinson's Disease* SCIENTIFIC RePoRTS, 2018. **8**.
11. Bengio, Y., et al. *Greedy layer-wise training of deep networks*. in *Advances in Neural Information Processing Systems 19*. 2007. MIT Press.
12. Catchpoole, D.R., et al., *The curse of dimensionality: a blessing to personalized medicine.* J Clin Oncol, 2010. **28**(34): p. e723-4; author reply e725.
13. Fakoor, R., et al. *Using deep learning to enhance cancer diagnosis and classification*. in *The 30th International Conference on Machine Learning (ICML 2013)*. 2013.
14. Aliferis, C.F., et al. *Machine Learning Models for Classification of Lung Cancer and Selection of Genomic Markers Using Array Gene Expression Data*. in *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference*. 2003. St. Augustine, Florida, USA.
15. Sharma, A., S. Imoto, and S. Miyano, *A top-r feature selection algorithm for microarray gene expression data.* IEEE/ACM Trans Comput Biol Bioinform, 2012. **9**(3): p. 754-64.
16. Moon, M. and K. Nakai, *Stable feature selection based on the ensemble L.* BMC Genomics, 2016. **17**(Suppl 13): p. 1026.
17. Degenhardt, F., S. Seifert, and S. Szymczak, *Evaluation of variable selection methods for random forests and omics data sets.* Brief Bioinform, 2017.
18. Higham, C.F., et al., *Deep learning for real-time single-pixel video.* Sci Rep, 2018. **8**(1): p. 2369.
19. Collobert, R., et al., *Natural Language Processing (Almost) from Scratch.* Journal of Machine Learning Research, 2011. **12**: p. 2493-2537.

20. Lenz, I., H. Lee, and A. Saxena, *Deep learning for detecting robotic grasps.* The International Journal of Robotics Research, 2015. **34**(4-5): p. 705-724.
21. Levine, S., et al., *Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection.* The International Journal of Robotics Research, 2017.
22. LeCun, Y., Y. Bengio, and G. Hinton, *Deep learning.* Nature, 2015. **521**(7553): p. 436-44.
23. Wang, X., et al., *Searching for prostate cancer by fully automated magnetic resonance imaging classification: deep learning versus non-deep learning.* Sci Rep, 2017. **7**(1): p. 15415.
24. Bychkov, D., et al., *Deep learning based tissue analysis predicts outcome in colorectal cancer.* Sci Rep, 2018. **8**(1): p. 3395.
25. Alipanahi, B., et al., *Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning.* Nat Biotechnol, 2015. **33**(8): p. 831-8.
26. Zhou, J. and O.G. Troyanskaya, *Predicting effects of noncoding variants with deep learning-based sequence model.* Nat Methods, 2015. **12**(10): p. 931-4.
27. Nair, V. and G.E. Hinton. *Rectified linear units improve restricted boltzmann machines*. in *the 27th International Conference on Machine Learning*. 2010. Haifa, Israel.
28. Larochelle, H., et al., *Exploring Strategies for Training Deep Neural Networks.* Journal of Machine Learning Research, 2009. **1**: p. 1-40.
29. Hinton, G.E., S. Osindero, and Y.W. Teh, *A fast learning algorithm for deep belief nets.* Neural Comput, 2006. **18**(7): p. 1527-54.
30. Hinton, G.E. and R.R. Salakhutdinov, *Reducing the dimensionality of data with neural networks.* Science, 2006. **313**(5786): p. 504-7.
31. Tan, J., et al., *Unsupervised feature construction and knowledge extraction from genome-wide assays of breast cancer with denoising autoencoders.* Pac Symp Biocomput, 2015: p. 132-43.
32. Gupta, A., H. Wang, and M. Ganapathiraju. *Learning structure in gene expression data using deep architectures, with an application to gene clustering*. in *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2015. Washington, DC, USA: IEEE.
33. Nair, V. and G. Hinton, *3D Object Recognition with Deep Belief Nets*, in *Advances in Neural Information Processing Systems 22*, Y. Bengio, et al., Editors. 2009, Curran Associates, Inc. p. 1339-1347.
34. Lee, H., C. Ekanadham, and A. Ng, *Sparse deep belief net model for visual area V2*, in *Advances in Neural Information Processing Systems 20*. 2008, Curran Associates, Inc. p. 873-880.
35. Le, Q., et al. *Building High-level Features Using Large Scale Unsupervised Learning* in *the 29th International Coference on International Conference on Machine Learning*. 2012. Edinburgh, Scotland: Omnipress, USA.
36. Deng, J., et al. *Sparse autoencoder-based feature transfer learning for speech emotion recognition*. in *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*. 2013. IEEE.
37. Larochelle, H., et al. *An Empirical Evaluation of Deep Architectures on Problems with Many Factors of Variation*. in *the 24th International Conference on Machine Learning*. 2007. Corvallis, OR: ACM New York, NY , USA.

38. Vincent, P., et al., *Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion.* Journal of Machine Learning Research 2010. **11**: p. 3371-3408.
39. Furey, T.S., et al., *Support vector machine classification and validation of cancer tissue samples using microarray expression data.* Bioinformatics, 2000. **16**(10): p. 906-14.
40. Khan, J., et al., *Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks.* Nat Med, 2001. **7**(6): p. 673-9.
41. Pal, N.R., et al., *Discovering biomarkers from gene expression data for predicting cancer subgroups using neural networks and relational fuzzy clustering.* BMC Bioinformatics, 2007. **8**: p. 5.
42. Chang, K.H., et al., *MicroRNA signature analysis in colorectal cancer: identification of expression profiles in stage II tumors associated with aggressive disease.* Int J Colorectal Dis, 2011. **26**(11): p. 1415-22.
43. Nanni, L., S. Brahnam, and A. Lumini, *Combining multiple approaches for gene microarray classification.* Bioinformatics, 2012. **28**(8): p. 1151-7.
44. Jordan, D.M. and R. Do, *Using Full Genomic Information to Predict Disease: Breaking Down the Barriers Between Complex and Mendelian Diseases.* Annu Rev Genomics Hum Genet, 2018.
45. Hathaway, R. and J. Bezdek, *NERF c-Means: Non-Euclidean relational fuzzy clustering.* Pattern Recognition, 1994. **27**: p. 429-437.
46. Danaee, P., R. Ghaeini, and D.A. Hendrix, *A DEEP LEARNING APPROACH FOR CANCER DETECTION AND RELEVANT GENE IDENTIFICATION.* Pac Symp Biocomput, 2016. **22**: p. 219-229.
47. Singh, V., et al. *Layerwise feature selection in Stacked Sparse Auto-Encoder for tumor type prediction*. in *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2016.
48. Powers, D.M.W., *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation.* Journal of Machine Learning Technologies, 2011. **2**(1): p. 37-63.
49. Stiglic, G. and P. Kokol, *Stability of ranked gene lists in large microarray analysis studies.* J Biomed Biotechnol, 2010. **2010**: p. 616358.
50. Hagan, M., et al., *Generalization*, in *Neural Network Design*. 2015. p. 468-519.
51. Bergstra, J. and Y. Bengio, *Random Search for Hyper-Parameter Optimization.* Journal of Machine Learning Research, 2012. **13**: p. 281-305.
52. Snoek, J., H. Larochelle, and R. Adams. *Practical Bayesian Optimization of Machine Learning Algorithms*. in *the 25th International Conference on Neural Information Processing Systems*. 2012. Lake Tahoe, Nevada: Curran Associates Inc., USA.
53. He, H. and E.A. Garcia. *Learning from Imbalanced Data*. in *IEEE Transactions on Knowledge and Data Engineering*. 2009. IEEE.
54. Johnson, W.E., C. Li, and A. Rabinovic, *Adjusting batch effects in microarray expression data using empirical Bayes methods.* Biostatistics, 2007. **8**(1): p. 118-27.
55. Shendure, J. and H. Ji, *Next-generation DNA sequencing.* Nat Biotechnol, 2008. **26**(10): p. 1135-45.
56. Rasmus, A., et al. *Semi-supervised learning with Ladder networks*. in *the 28th International Conference on Neural Information Processing Systems* 2015. Montreal, Canada: MIT Press Cambridge, MA, USA.

730 57. Cutler, M. and J.P. How. *Efficient Reinforcement Learning for Robots Using Informative*
731 *Simulated Priors*. in *2015 IEEE International Conference on Robotics and Automation*
732 *(ICRA)*. 2015. Seattle, WA, USA: IEEE.
733 58. Mnih, V., et al., *Human-level control through deep reinforcement learning.* Nature, 2015.
734 **518**(7540): p. 529-33.
735 59. Ching, T., et al., *Opportunities And Obstacles For Deep Learning In Biology And*
736 *Medicine*. 2017: bioRxiv.
737