A Sparse Convolutional Predictor with Denoising Autoencoders for Phenotype Prediction

Junjie Chen
University of North Carolina at Charlotte
Charlotte, North Carolina, USA
junjie.chen.hit@gmail.com

Xinghua Shi*
University of North Carolina at Charlotte
Charlotte, North Carolina, USA
mindyshi@gmail.com

ABSTRACT

Phenotype prediction has been widely conducted in many areas to help understand disease risks and susceptibility, and improve the breeding cycles of plants and animals. Most methods of phenotype prediction are based on regularized statistical approaches which only consider linear relationships among genetic features. Deep learning based methods have been recently reported to nicely address regression problems in high dimensional data in genomic studies. To explore deep learning for phenotype prediction, we propose a deep learning regression model, called Sparse Convolutional Predictor with Denoising Autoencoders (SCP_DAE), to predict quantitative traits. We constructed SCP_DAE by utilizing a convolutional layer that can extract correlation or linkage patterns in the genotype data and applying a sparse weight matrix resulted from the L_1 regularization to handle high dimensional genotype data. To learn efficient and compressed hidden representations of genotype data, we pre-trained the convolutional layer and the first fully connected layer in SCP_DAE using denoising autoencoders. These pre-trained layers were then fine-tuned to improve its performance of the SCP_DAE model for phenotype prediction. We comprehensively evaluated our proposed method on a yeast dataset which contains well assayed genotype profiles and quantitative traits. Our results showed that the proposed SCP_DAE method significantly outperforms regularized statistical approaches and similar deep learning models without pre-trained weights.

CCS CONCEPTS

• **Applied computing** → **Computational genomics**.

KEYWORDS

deep learning; convolutional network; autoencoder; sparse model; phenotype prediction; genomics

ACM Reference Format:
Junjie Chen and Xinghua Shi. 2019. A Sparse Convolutional Predictor with Denoising Autoencoders for Phenotype Prediction. In *10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics (ACM-BCB '19), September 7–10, 2019, Niagara Falls, NY, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3307339.3342179

ACM-BCB '19, September 7–10, 2019, Niagara Falls, NY, USA

ACM ISBN 978-1-4503-6666-3/19/09…$15.00
https://doi.org/10.1145/3307339.3342179

1 INTRODUCTION

Recent advances in biotechnology, particularly second- and third-generation sequencing, have made genomics data available at an unprecedented scale [1]. The abundant genomic data consequently demand new methods that leverage powerful computing to address many classical yet challenging problems in genomics and genetics. One of such problems is phenotype prediction, that aims to investigate how genetic variation among individuals, can be used to predict phenotypic differences in these individuals for a particular phenotype such as disease risks or the yield of a biomass of interest. Phenotype prediction methods have been reported to help dissect genetic causes and risks of complex phenotype including common diseases, and improve the breeding cycles of plants and animals [14]. However, there are several barriers that make it extremely challenging to develop robust and powerful methods for phenotype prediction. First, genomic data is typically high dimensional, where the number of features p is significantly larger than the sample size n (denoted the '$n \ll p$' problem). Second, genomic data is usually with high collinearity and correlation that introduces intrinsic patterns or structures to the data.

Many methods have been developed for phenotype prediction including linear mixed models (LMMs), regularized statistic approaches and machine learning approaches. LMMs are widely used in genetic selection prediction, including Best Linear Unbiased Prediction (BLUP) based algorithms, like ridge regression BLUP (rrBLUP) [14] and genomic relationship BLUP (GBLUP) [25]. Applications of LMMs to polygenic modeling assume that every genetic variant affects the phenotype with effect sizes normally distributed, and predict phenotypes from a linear function of genetic markers. Regularized statistic methods like least absolute shrinkage and selection operator (Lasso) [24] have been proposed for phenotype prediction using shrinkage and variable selection operators. A number of Lasso variants have been developed to remedy certain limitations of the original technique and to make the method more useful for particular problems. These methods include elastic nets [30] that add an additional ridge regression-like penalty to improve performance when the number of genetic variants is significantly larger than the sample size. Another approach is a spike-and-slab Lasso (ssLasso) method [23] that addresses the limitations of Lasso by imposing weak or no shrinkage on related features while imposing strong shrinkage on unrelated features. Bayesian-based methods are proposed for phenotype prediction as well, which includes a hybrid of LMMs and sparse regression models such as Bayesian Lasso [17] and Bayesian sparse linear mixed model (BSLMM) [29]. In addition to achieving variable selection like the Lasso based methods, Bayesian approaches are suitable for coefficient estimates and statistical assessment of uncertainty [18].

Machine learning methods for phenotype prediction include traditionally models like random forest [8] and support vector machine (SVM) [26]. Recently, deep learning has emerged as a powerful machine learning method that builds multi-layered neural networks to model complex relationship in big data [10]. Deep learning has gained tremendous attention in bioinformatics due to its outstanding performances in many tasks [15], including the gene variants and expressions [4, 21, 28], protein structure prediction [7, 11], etc. Several studies have reported that deep learning has strong potentials to address high dimensional regression problems. For example, a convolutional network was developed to predict the overall survival of patients diagnosed with brain tumors from microscopic images of tissue biopsies and genomic biomarkers [16]. Particularly, a new deep learning method called DeepGS was proposed that uses a deep convolutional neural network (CNN) to explore the hidden genetic architecture for phenotype prediction [13]. Another deep learning model [12] was constructed that uses a CNN to predict genomic estimated breeding value and also to investigate neighboring SNP effects with linkage disequilibrium (LD) [20]. Deep learning can solve the regression problem by learning an efficient representation of input. However, it is still a challenging problem to learn an efficient representation on high dimensional data. One particular technique to learn an efficient representation of data is to pre-train the weights by using autoencoders (AE) [3] or denoising autoencoder (DAE) [27]. Both AE and DAE are unsupervised artificial neural networks that are designed to learn efficient data coding and reconstruct the input. Hence, deep learning models including CNNs and DAE are worthwhile for further investigation to support genome-wide phenotype prediction on high dimensional genomic data.

In this study, we propose a Sparse Convolutional Predictor with Denoising Autoencoders (SCP_DAE) for phenotype prediction. SCP_DAE can utilize the sparse convolution layer to captures linkage patterns in high dimensional genomic data, and leverage pre-trained weights by DAE to learn an efficient and compressed representation of input data. We comprehensively evaluated the performance of our proposed model on real yeast genomic data. Our results show that SCP_DAE achieved significantly lower mean squared errors than existing methods such as Lasso, elastic net, and random forest. This study thus points to a direction in phenotype prediction and other challenging problems in genomic data analysis, where sparse convolutional networks are coupled with denoised autoencoders to significantly improve model performance.

2 MATERIALS AND METHODS

2.1 Dataset

We employed a comprehensively assayed yeast dataset [2] to evaluate the proposed SCP_DAE method. This dataset contains genotype profiles of 28,820 unique genetic variants that were obtained by sequencing 4,390 individuals from a cross between two strains of yeast: a widely used laboratory strain (BY) and an isolate from a vineyard (RM). The original data fields in the yeast genotype profiles were encoded as -1 for BY and 1 for RM. Since the loss function in our proposed model requires non-negative data fields, we replaced all -1 values with 2 when preprocessing the genotype data. Together with the profiled genotypes, this yeast population was phenotyped for 20 end-point growth traits with at least two replicates. One important characteristic about this genotype data is that it includes epistasis, or non-linear interactions, among genetic variants that are not easy to model in a typical method for phenotype prediction. In this study, we picked three phenotypes for investigation including Cobalt Chloride, Copper Sulfate, and Diamide. For each trait, we calculated the mean of two replicates as the final phenotypes. All phenotype values were normalized while the NAs were ignored and outliers were removed.

As a commonly-used way to pre-process data in machine learning, we randomly split the whole dataset into three separate subdatasets containing 80%, 10% and 10% for training, validation and testing respectively. The missing values in the input of denoising autoencoder were generated by randomly masking 10% of the original genotypes to zeros when training denoising autoencoder models. All methods for comparison were run repeatedly for 10 times on 10 random splits.

2.2 Sparse convolutional network

To take into account of the intrinsic correlation or linkage patterns of genotype data, we leverage convolution networks to learn the underlying structures and relationships in data. Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of an animal's visual cortex [9]. Each convolutional neuron processes data only for its receptive field. Such character of convolutional network makes it able to successfully capture the spatial or local patterns in genotype data, such as the highly correlated patterns such as LD [20] in genotype data.

Since genotype values are discrete, we use convolution operations in a discrete space as defined in Eq. (1).

$$O(i) = \sum_{u=1}^{k} F(u)I(i-u) \tag{1}$$

where $O(i)$ is the output of the i marker in input. F is the convolutional filter. k is the convolutional filter size, represented as an odd number. I is the input vector. The convolution operation is done for every location of the input vector I and thus for each genetic variant, that completely overlaps with the convolutional filter.

Every convolutional layer is composed of n convolutional filters, each with depth D, where D is the input depth. A convolution among an input $I = \{I_1, \cdots, I_D\}$ and a set of n convolutional filters $\{F_1, \cdots, F_n\}$, each with depth D, produces a set of n activation maps , or equivalently, a volume of activation maps with depth n based on Eq. (2).

$$O_m = \sigma(I \otimes F_m + b_m) \quad m = 1, \cdots, n \tag{2}$$

where σ is a non-linear activation function, $\otimes$ is a convolution symbol of Eq. (1). Here, b_m is the bias, and m represents the m^{th} feature map.

To prevent overfitting and introducing model sparsity for analyzing high-dimensional genomic data, we use sparse convolutional networks by introducing the L_1 regularization to all convolutional filters. More specifically, the L_1 regularization is conducted by applying L_1 penalties to layer weights during the optimization process of the model. These penalties are incorporated in the loss function where the model will optimize on. The L_1 regularization, described

in Eq. (3), will penalize or shrink small weights to zeros to improve the robustness and sparsity of the model.

$$L_1 = \lambda \sum_{m=1}^{n} \|F_m\|_{1,1} \tag{3}$$

where $\|\cdot\|$ refers to the $L_{1,1}$ norm of matrix, F_m is m^{th} convolution filter weight matrix in the layer, and $\lambda \in [0, 1]$ is a hyperparameter to control the shrinkage. The larger λ leads to the more sparse model.

2.3 Autoencoder and denoising autoencoder

An autoencoder (AE) [3] is an unsupervised artificial neural network that is designed to learn efficient data coding to reconstruct the input data. As shown in **Fig. 1.a**, an autoencoder consists of two parts: an encoder and a decoder, which can be defined as f and g respectively. The encoder takes an input vector $\mathbf{x} \in \Re^n$ and maps it to a hidden representation $\mathbf{h} \in \Re^m$ through a mapping function in Eq. (4).

$$\mathbf{h} = f_\theta(\mathbf{x}) = \Phi(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{4}$$

where $\theta = \{\mathbf{W}, \mathbf{b}\}$, $\mathbf{W}$ is a $m \times n$ weight matrix, $\mathbf{b}$ is a bias vector and Φ is an activation function such as a sigmoid [6] or Rectified Linear Units (ReLU) [5]. The hidden representation $\mathbf{h}$ is also called a latent representation. The decoder takes a hidden representation $\mathbf{h}$ and map it to a reconstructed input vector $\mathbf{z} \in \Re^n$ using Eq. (5).

$$\mathbf{z} = g_{\theta'}(\mathbf{h}) = \Phi'(\mathbf{W}'\mathbf{h} + \mathbf{b}') \tag{5}$$

where $\theta' = \{\mathbf{W}', \mathbf{b}'\}$, $\mathbf{W}'$ is a $n \times m$ weight matrix, $\mathbf{b}'$ is a bias vector and Φ' is an activation function, the same as Φ. The parameters θ and θ' of an autoencoder will be optimized to minimize the average reconstruction error as shown in Eq. (6).

$$\begin{aligned} \theta^*, \theta'^* &= \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} L(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}) \\ &= \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} L(\mathbf{x}^{(i)}, g_{\theta'}(f_\theta(\mathbf{x}^{(i)}))) \end{aligned} \tag{6}$$

where θ^* and θ'^* are parameters to be learned on data.

The aim of an autoencoder is to reconstruct $\mathbf{z}^{(i)}$ such that $\mathbf{z}^{(i)} \approx \mathbf{x}^{(i)}$ by minimizing the loss function L which can be defined as the widely used mean squared error for continuous data or cross-entropy for discrete data. We minimized the binary cross-entropy loss using Eq. (7) between the input $\mathbf{x}$ and reconstructed $\mathbf{z}$, because the genotype values are discrete, particularly binary in the yeast genotype data.

$$L(\mathbf{x}, \mathbf{z}) = -(ylog(p) + (1 - y)log(1 - p)) \tag{7}$$

where y is a binary indicator (0 or 1) in the input $\mathbf{x}$, and p is the predicted probability in $\mathbf{z}$.

A denoising autoencoder (DAE) (**Fig. 1b**) [27] is an extension of a standard autoencoder, which differs in that a denoising autoencoder reconstructs the output from randomly corrupted data based on Eq. (8).

$$\theta^*, \theta'^* = \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} L(\mathbf{x}^{(i)}, g_{\theta'}(f_\theta(\mathbf{x}^{(i)}_{missing}))) \tag{8}$$

where θ^* and θ'^* are parameters to be learned on data. $\mathbf{x}$ represents original input, and $\mathbf{x}_{missing}$ represents corrupted data. The aim

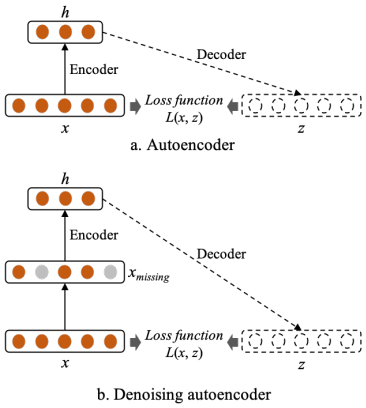

Figure 1: An illustration of a (a.) standard autoencoder and (b.) a denoising autoencoder.

of denoising autoencoder is to reconstruct $\mathbf{x}$ from $\mathbf{x}_{missing}$ by minimizing the loss function L.

Since a denoising autoencoder reconstructs the output from randomly corrupted data, it can let the encoder learn a more robust representation of the input data by preventing complex node interactions. This corruption process can be done by adding random noises to the input and optimize the autoencoders to remove the noises. In our experiments, the complete yeast data was corrupted by randomly masking 10% of original values to zeros, and then a denoising autoencoder model was trained on the corrupted data.

2.4 The proposed method

We constructed a sparse convolutional predictor (SCP) (**Fig. 2a**) for predicting yeast phenotypes. Convolutional networks are used to incorporate local correlation or linkage patterns from input data. Each convolutional kernel generates a feature map from the input, and in this process, correlation patterns in the filtering window of the convolutional layer can be learned. Moreover, we introduce an L_1 regularization to every layer to induce sparsity in the model. The L_1 regularization is set as 1E-5. Regression layers are fully connected layers which can capture the global dependencies that are distant from each other. For predicting a quantitative trait, the last two layers are fully connected layers with the dimension of one quarter and one tenth of original dimension, respectively, with the output as a real number. In this study, the hyperparameters were empirically optimized on training and validation dataset to predict the quantitative trait of Cobalt Chloride. The SCP has one convolutional layer with 8 convolutional kernels and a filter size of 5, followed by an average pooling layer with filter size of 2 and a dropout layer with drop percentage of 25%.

Since the feature dimension is much larger than sample size in genomic data, it is a challenging problem to learn an efficient representation for phenotype prediction. To address this challenge with an aim of learning an efficient representation and compressing the dimension of the genomic data, we use an AE and a DAE to pre-train the weights of a convolutional layers. The AE and DAE have same architecture, but the input of DAE contains noises. **Fig. 2b** shows the architecture of a DAE for pre-training the weights.

The encoder in a DAE has the same hyperparameters with a convolutional layer in SCP. The compressed representation is the same with the first fully connected layer in SCP. The decoder in a DAE is a mirror of the encoder. An AE takes the original genotype data to reconstruct itself. A DAE takes the corrupted genotype data to reconstruct the original genotype data. In these processes, the weights of the encoder are trained to learn an efficient representation. At last, the trained weights in AE or DAE will need to fine tune in the SCP in order to perform phenotype prediction.

In summary, the weights of a convolution layer and the first fully connected layer in SCP are first pre-trained by using a AE or DAE. Then the SCP is initialized with pre-trained weights. Finally, all layers in the SCP architecture are fine tuned.

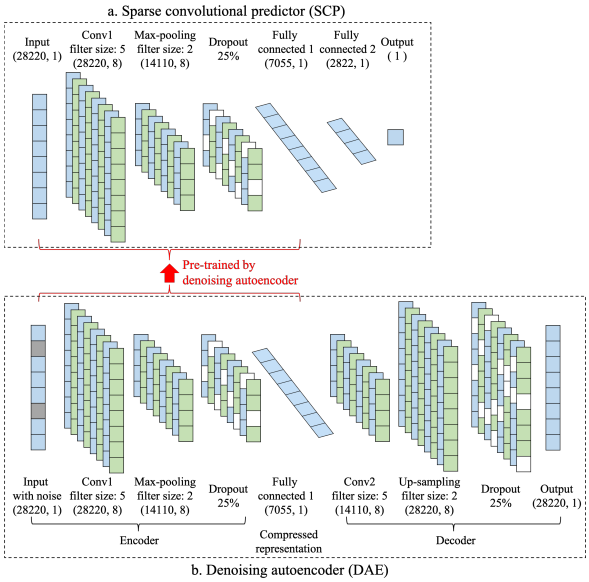

Figure 2: The architecture of our proposed sparse convolutional predictor with denoising autoencoders.

3 RESULTS

3.1 The sparsity of the weight matrix

We investigated the effect of sparsity of weight matrix on training and validation dataset to predict the quantitative trait of Cobalt Chloride, in terms of mean squared errors between predicted and observed phenotype values. **Fig. 3** shows the effect of different L_1 norm regularization values. Overfitting obviously occurred while no regularization was applied, while it was was restricted when the regularization value was 1E-6. However, if the regularization is too strong, the predictive performance can decrease as shown in the result of a large regularization of 1E-4. Hence, we need to find a balance between overfitting and model performance when choosing a regularization hyperparameter. In this study, SCP achieved the best performance when L_1 norm regularization value was 1E-5. Not only did the overfitting disappear, but also was the predictive performance improved when appropriate regularization was used in the SCP model.

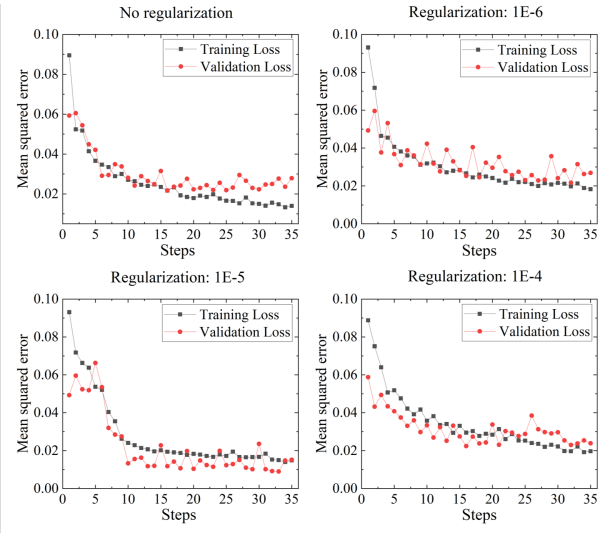

Figure 3: The effect of L_1 regularization.

The kernel visualization of a convolution layer in SCP with regularization of 1E-5 is shown in **Fig. 4**. There are 8 kernels with a size of 5 in the convolution layer. The squares in green indicate positive weights, and those in magenta represent negative weights. The deeper the color is, the larger absolute value the weight is. Grey squares represent weights with zero values, indicating the sparsity of our model.

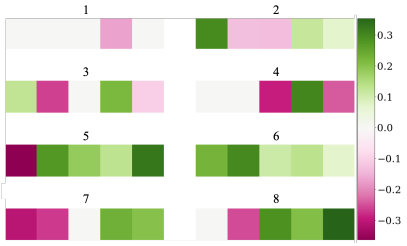

Figure 4: The kernel visualization of the convolution layer in SCP with a regularization of 1E-5.

3.2 Pre-trained weights by autoencoders

Although L_1 regularization can improve robustness and prevent overfitting, the efficient and compressed representations of yeast samples are hard to learn due to the high dimension of the yeast genotype data. Thus, we introduce a denoising autoencoder to learn efficient representations of yeast samples. **Fig. 2b** shows the architecture of a denoising autoencoder used in this study. In comparison, we also include a standard authoencoder in the model. The only one difference between an autoencoder and a denoising autoencoder is that 10% genotypes were masked as zeros in training a denoising autoencoder.

We compared the performance between an autoencoder and a denoising autoencoder in terms of average accuracy for reconstructing original data for 10 random repeats by randomly splitting the

dataset. **Fig. 5** shows the violin plot for comparing the accuracy between an autoencoder and a denoising autoencoder. The accuracy refers to the percentage of correct genotypes of genetic variants in reconstructed yeast genotypes. A higher accuracy indicates that there is less loss between the original input and reconstructed output, and the compressed hidden representation contains more exact information from the original input. The violin plot for 10 repeated experiment results hence shows that denoising autoencoders outperform autoencoders in terms of average accuracy. This indicates that denoising autoencoder can extract efficient and robust features in yeast genotype, and the compressed representations learned by denoising autoencoders contain more exact information than autoencoders. Thus, we can expect the performance of SCP with denoising autoencoders outperforms the SCP with autoencoders as described later.

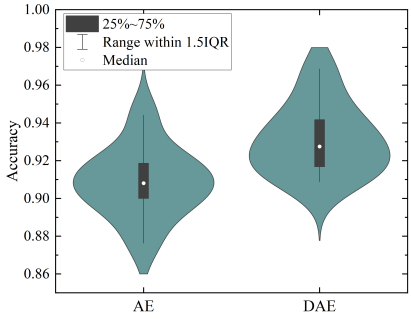

Figure 5: The model performance of an autoencdoer and denoising autoencoder in terms of accuracy.

3.3 The performance of sparse convolutional predictor

The performance of SCP, SCP_AE (SCP with pre-trained weights by using autoencoders), and SCP_DAE (SCP with pre-trained weights by using denoising autoencoders) was listed in **Table 1**. We observed that SCP_DAE achieved the mean square errors of 0.0051, 0.0024, and 0.0049 for three quantitative traits of Cobalt Chloride, Copper Sulfate, and Diamide in yeast, respectively. Our final model of SCP_DAE thus outperformed SCP_AE, which had mean square errors of 0.059, 0.038 and 0.066 for the three quantitative traits repectively. This nice performance of SCP_DAE thus comes from the characteristics of a denoising autoencoder in that it reconstructs the output from the corrupted input data to let the encoder extract most important features and learn a more robust representation of the original input data.

Fig. 6 shows the predictive results of SCP_DAE on Cobalt Chloride trait of some yeast samples in the test dataset. The predicted results aligned well with most of observed yeast traits, and they had similar peaks and changing trends to the assayed phenotypes.

3.4 Comparison with other methods

We compared our proposed SCP_DAE method, a straight SCP, and SCP_AE method, with other popular regularized statistical methods and machine learning methods for phenotype prediction including Lasso, elastic net, random forest, and a plain deep neural

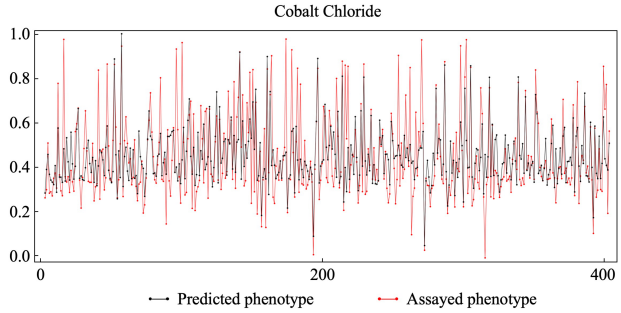

Figure 6: Visualization of the assayed phenotype (in red) and the predicted phenotype (in black) by SCP_DAE on test data.

network (DNN). Lasso and elastic net are popular regularization-based methods for sparsity learning, and random forest is one of popular machine learning methods for regression problems. All of these methods are widely used in high dimensional genomic data analysis. For Lasso, the hyperparameter λ is learned via cross-validation. For elastic net, the hyperparameter α that controls the balance of L_1 and L_2 norm was set as a conventional value 0.5, and the hyperparameter λ that controls the model sparsity was optimized via cross-validation, similar to Lasso. For random forest, all hyperparameters were automatically decided depending on the genotype dimension as default in the randomForest package in R. DNN has three fully connected layers with the number of neural nodes of 14110, 7055 and 2822, respectively, comparing with our proposed SCP in that DNN does not have convolutional layers.

As shown in **Table 1**, Lasso, elastic net and random forest have similar performance on the three quantitative traits. DNN performs slightly worse than previous methods, because the fully connected layers in DNN are hard to capture the underlying local linkages and local genetic loci dependencies in genomic data and thus it fails for phenotype prediction. The straightforward SCP slightly outperforms Lasso, elastic net and random forest. Comparing with regularization-based methods, SCP not only leverages L_1 regularization to learn sparse weights, but also it can extract the non-linear relationship between genetic loci dependencies. Comparing with DNN, SCP utilizes convolution layers to learn the local linkages and employs the followed fully connected layers to capture global genetic loci dependencies.

The performance of SCP is significantly improved by pre-training weights in an autoencoder and a denoising autoencoder. SCP_AE achieved mean square errors of 0.059, 0.038 and 0.066 for the three quantitative traits respectively. SCP_DAE further improves the performance of SCP_AE with mean squared errors of 0.0051, 0.0024, and 0.0049 for the three traits respectively. Such nice results of SCP_DAE reflect the benefits from the pre-trained weights using denoising autoencoders.

4 CONCLUSION

In summary, we have proposed a novel sparse convolutional predictor with denoising autoencoder for phenotype prediction. Our method leverages a convolutional layer to learn the underlying

Table 1: Performance comparison of different methods on predicting three yeast quantitative traits in terms of average mean squared errors over 10 repeats.

Methods	Phenotypes		
	Cobalt Chloride	Copper Sulfate	Diamide
Lasso	0.0118	0.0070	0.0092
Elastic net	0.0115	0.0074	0.0089
Random forest	0.0114	0.0073	0.0121
DNN	0.0118	0.0117	0.0164
SCP	0.0109	0.0059	0.0113
SCP_AE	0.0059	0.0038	0.0066
SCP_DAE	**0.0051**	**0.0024**	**0.0049**

correlation or linkage patterns in input data, and applies L_1 regularization to learn sparse weights on high dimensional data. The process of pre-training weights using a denoising autoencoder further improves the model's predictive performance. Our newly proposed method achieves state-of-the-art performance comparing with other popular regularized statistic methods and machine learning methods on a comprehensive yeast dataset. This study thus demonstrates that sparse convolutional networks coupled with pre-trained weights by denoising autoencoders can significantly improve prediction tasks on high dimensional genotype data.

There are several directions that we can explore in the future to improve the proposed model. It is a key problem to learn an efficient representation of input data to build a powerful and robust phenotype prediction method. Hence, it should be interesting to explore other effective architectures for pre-training layer weights such as generative adversarial network [19]. To induce sparsity to the model, in addition to the L_1 norm, other regularization can be explored such as the L_1-L_2 norm. Bayesian optimization strategies [22] can be employed to improve hyperparameter tuning in the model. Although our method is designed for quantitative traits prediction, it can be extended to categorical traits by slightly changing the output layer.

ACKNOWLEDGMENTS

This work is partially supported by National Science Foundation of the United States (Award Number: 1750632).

REFERENCES

[1] Anna Alemany, Maria Florescu, Chloé S Baron, Josi Peterson-Maduro, and Alexander Van Oudenaarden. 2018. Whole-organism clone tracing using single-cell sequencing. *Nature* 556, 7699 (2018), 108.

[2] Joshua S Bloom, Iulia Kotenko, Meru J Sadhu, Sebastian Treusch, Frank W Albert, and Leonid Kruglyak. 2015. Genetic interactions contribute less than additive effects to quantitative trait variation in yeast. *Nature communications* 6 (2015), 8712.

[3] Lujia Chen, Chunhui Cai, Vicky Chen, and Xinghua Lu. 2016. Learning a hierarchical representation of the yeast transcriptomic machinery using an autoencoder model. *BMC Bioinformatics* 17, 1 (11 Jan 2016), S9. https://doi.org/10.1186/s12859-015-0852-1

[4] Yifei Chen, Yi Li, Rajiv Narayan, Aravind Subramanian, and Xiaohui Xie. 2016. Gene expression inference with deep learning. *Bioinformatics* 32, 12 (2016), 1832–1839.

[5] George E Dahl, Tara N Sainath, and Geoffrey E Hinton. 2013. Improving deep neural networks for LVCSR using rectified linear units and dropout. In *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 8609–8613.

[6] Jun Han and Claudio Moraga. 1995. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *International Workshop on Artificial Neural Networks*. Springer, 195–201.

[7] Rhys Heffernan, Yuedong Yang, Kuldip Paliwal, and Yaoqi Zhou. 2017. Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility. *Bioinformatics* 33, 18 (2017), 2842–2849.

[8] Jason A Holliday, Tongli Wang, and Sally Aitken. 2012. Predicting adaptive phenotypes from multilocus genotypes in Sitka spruce (Picea sitchensis) using random forest. *G3: Genes, Genomes, Genetics* 2, 9 (2012), 1085–1093.

[9] David H Hubel and Torsten N Wiesel. 1968. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology* 195, 1 (1968), 215–243.

[10] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.

[11] Haiou Li, Jie Hou, Badri Adhikari, Qiang Lyu, and Jianlin Cheng. 2017. Deep learning methods for protein torsion angle prediction. *BMC bioinformatics* 18, 1 (2017), 417.

[12] Yang Liu and Duolin Wang. 2017. Application of deep learning in genomic selection. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2280–2280.

[13] Wenlong Ma, Zhixu Qiu, Jie Song, Jiajia Li, Qian Cheng, Jingjing Zhai, and Chuang Ma. 2018. A deep convolutional neural network approach for predicting phenotypes from genotypes. *Planta* 248, 5 (2018), 1307–1318.

[14] T.H.E. Meuwissen, B.J. Hayes, and M.E. Goddard. 2001. Prediction of total genetic value using genome-wide dense marker maps. *Genetics* 157, 4 (2001), 1819–1829.

[15] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. 2017. Deep learning in bioinformatics. *Briefings in bioinformatics* 18, 5 (2017), 851–869.

[16] Pooya Mobadersany, Safoora Yousefi, Mohamed Amgad, David A Gutman, Jill S Barnholtz-Sloan, José E Velázquez Vega, Daniel J Brat, and Lee AD Cooper. 2018. Predicting cancer outcomes from histology and genomics using convolutional networks. *Proceedings of the National Academy of Sciences* 115, 13 (2018), E2970–E2979.

[17] Trevor Park and George Casella. 2008. The bayesian Lasso. *J. Amer. Statist. Assoc.* 103, 482 (2008), 681–686.

[18] Menelaos Pavlou, Gareth Ambler, Shaun Seaman, Maria De Iorio, and Rumana Z Omar. 2016. Review and evaluation of penalised regression methods for risk prediction in low-dimensional data with few events. *Statistics in medicine* 35, 7 (2016), 1159–1177.

[19] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[20] David E Reich, Michele Cargill, Stacey Bolk, James Ireland, Pardis C Sabeti, Daniel J Richter, Thomas Lavery, Rose Kouyoumjian, Shelli F Farhadian, Ryk Ward, et al. 2001. Linkage disequilibrium in the human genome. *Nature* 411, 6834 (2001), 199.

[21] Ritambhara Singh, Jack Lanchantin, Gabriel Robins, and Yanjun Qi. 2016. DeepChrome: deep-learning for predicting gene expression from histone modifications. *Bioinformatics* 32, 17 (2016), i639–i648.

[22] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. *arXiv:1206.2944 [cs, stat]* (June 2012). http://arxiv.org/abs/1206.2944 arXiv: 1206.2944.

[23] Zaixiang Tang, Yueping Shen, Xinyan Zhang, and Nengjun Yi. 2016. The Spike-and-Slab Lasso Generalized Linear Models for Prediction and Associated Genes Detection. *Genetics* (2016), genetics–116.

[24] Robert Tibshirani. 1996. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.

[25] Paul M VanRaden. 2008. Efficient methods to compute genomic predictions. *Journal of dairy science* 91, 11 (2008), 4414–4423.

[26] Vladimir Vapnik. 1998. *Statistical learning theory. 1998*. Vol. 3. Wiley, New York.

[27] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. ACM, 1096–1103.

[28] Jian Zhou, Chandra L Theesfeld, Kevin Yao, Kathleen M Chen, Aaron K Wong, and Olga G Troyanskaya. 2018. Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk. *Nature genetics* 50, 8 (2018), 1171.

[29] Xiang Zhou, Peter Carbonetto, and Matthew Stephens. 2013. Polygenic modeling with Bayesian sparse linear mixed models. *PLoS Genetics* 9, 2 (2013), e1003264.

[30] Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67, 2 (2005), 301–320.