A Deep Learning Method to Impute Missing Values and Compress Genome-wide Polymorphisms for Predicting Phenotype of Rice

Islam Tanzila, Graduate School of Science and Engineering, Iwate University

Abstract: Genomic selection (GS) is expected to accelerate plant and animal breeding, but during the last decade has seen a huge increase in genome-wide polymorphism data, raising concerns about storage costs and computational time. This necessitates the development of innovative platforms that can significantly reduce the resources required for storage and processing. Several individual studies, such as imputation of missing genomic data, compression of genomic data, and phenotype prediction, have attempted to address these concerns. However, some problems remain: imputation models are developed only for imputing missing genotypes, compression models lack adequate data quality after compression, and prediction models are time-consuming. Therefore, this research proposes a novel combined deep learning model that could overcome these limitations and outperform the other state-of-the-art methods. The proposed model is roughly composed of three parts: (i) AGI (autoencoder genome imputation) for imputing missing genome-wide polymorphisms, (ii) AGC (autoencoder genome compression) for compressing genome-wide polymorphism data by generating different levels of compression according to storage requirements, and (iii) DeepCGP (deep learning compression-based genomic prediction) for predicting phenotypes from compressed information using random forests, genomic best linear unbiased prediction, and Bayesian variable selection.

1. Introduction

Crop improvement is one of the essential means of bridging the gap between ever-growing human population and food shortage (Qaim, 2020). By 2050, 70% more food production is required to keep pace with the expected increase in food demand and ongoing climate change on a global scale (He and Li, 2020). To achieve this challenge, we need to enhance genetic gains in plant breeding through novel technologies (Tester and Langridge, 2010; Qaim, 2020). One such technology is the use of genome-phenotype associations (Hamblin *et al.*, 2011; Kole *et al.*, 2015; Thudi *et al.*, 2021). These include genome-wide association studies (GWAS), in which candidate genes are discovered based on the associations (Gupta *et al.*, 2014) and genomic selection (GS) (Meuwissen *et al.*, 2001), in which the genetic abilities of individuals or lines are predicted and selected. GS is expected to be effective in improving complex traits (e.g., crop yield) controlled by a large number of genes, which have been difficult to improve (Jannink *et al.*, 2010).

The use of genomic data is progressing in various fields, and a massive amount of genomic data has been generated (Stephens *et al.*, 2015) as a resource for plant breeding (Thudi *et al.*, 2021). Furthermore, with the introduction of high-throughput sequencing technologies, the number of data samples also tends to be large, resulting in challenges for storage and analysis of genomic data in the fields of genomics, bioinformatics, and quantitative genetics (Fan *et al.*, 2014). Moreover, the increasing size and dimension of data (Bhukya *et al.*, 2020) have led to an intensified need for data compression and compression-based data analysis. The ability to compress genomic data will not only make it easier to store and analyze data, but also aid in streamlining the exchange of data via Web APIs etc. (Selby *et al.*, 2019; Swaminathan *et al.*, 2016).

Since the last decade, researchers have been exploring various approaches to impute missing values (Gad *et al.*, 2020; Rana *et al.*, 2012). There have been many studies addressing the traditional genotype imputation methods, which are typically based on haplotype-clustering algorithms (Scheet and Stephens, 2006) and hidden Markov models (Marchini *et al.*, 2007). BEAGLE is another imputation tool based on a graphical model of a set of haplotypes (Browning and Browning, 2009; Browning *et al.*, 2018). It works iteratively by fitting the model to the current set of estimated haplotypes. Then resampling of new estimated haplotypes for each individual is conducted based on a fitted model. The probabilities of missing genotypes are calculated from the fitted model at the final iteration. Recently, deep learning based methods, especially autoencoders have shown great potential to impute missing values (Beaulieu-Jones and Moore, 2017; Duan *et al.*, 2016); for example Abdella and Marwala proposed a method to approximate missing data by using an autoencoder and genetic algorithm (Abdella and Marwala, 2005). In 2018, Lina proposed a Denoising Autoencoder with Partial Loss (DAPL) method to predict missing values in pan-cancer genomic analysis (Qiu *et al.*, 2018). They showed that the DAPL method achieves better performance with less computational burden over traditional imputation methods.

To effectively analyze high-dimensional data, deep learning (DL) techniques (Schmidhuber, 2015) have been introduced in various fields, including genomics, genetics, and breeding. Several DL methods exist (Absardi and Javidan, 2019; Wang *et al.*, 2019; Hinton and Salakhutdinov, 2006; Silva *et al.*, 2020) that can compress genomic data without compromising model performance. Wang et al. introduced DeepDNA to compress human mitochondrial genome data using hybrid convolutional and recurrent deep neural networks (Wang *et al.*, 2018). DeepDNA method has a good compression result in the population genome with large redundancy, and in a single genome with small redundancy. Goyal et al. introduced DeepZip, which used recurrent neural networks to compress genomics and text data (Goyal *et al.*, 2019). This method achieved a 20% better compression ratio than the Gzip compression algorithm (Deutsch, 1996).

For GS, accurate prediction of phenotypes (strictly speaking, genotypic values) of a target trait is a central and recurring problem in quantitative genetics. Consequently, several genomic prediction methods have been proposed based on machine learning (ML) (Szymczak *et al.*, 2009; Ogutu *et al.*, 2011, 2012; Okser *et al.*, 2014; González-Recio *et al.*, 2014; Blondel *et al.*, 2015) and quantitative genetic models, especially under a Bayesian paradigm (Meuwissen *et al.*, 2001; Gianola *et al.*, 2009; Habier *et al.*, 2011; Gianola, 2013). González et al. compared Bayes A and Bayesian LASSO with two machine learning algorithms (boosting and random forests [RFs]) to predict disease occurrence in simulated and real datasets (González-Recio and Forni, 2011). Although the differences between the methods were small, RF outperformed other methods in most cases. Abdollahi-Arpanahi et al. compared the predictive performance of two deep learning methods (multilayer perceptron [MLP] and convolutional neural network [CNN]), two ensemble learning methods (RF and gradient boosting), and two parametric methods (genomic best linear unbiased prediction [GBLUP] and BayesB) using real and simulated datasets (Abdollahi-Arpanahi *et al.*, 2020). The authors pointed out that the predictive performance of deep learning methods was marginally better than that of parametric methods for large datasets.

Generally, previously proposed methods in the literature had the following limitations: (i) imputing missing values from large datasets have severe drawbacks in computational overhead (ii) quality of information after the compression was uncertain, and (iii) original data was utilized for predictions using machine-learning methods. In contrast, in this study, the proposed method predicts phenotypes of target traits based on compressed genome-wide polymorphism data instead of original (i.e., uncompressed) data, which quantifies the quality of our compression method. Despite the compression of several cycles, the proposed method retains high-quality information, and the prediction accuracy of our method is similar to that of genomic prediction based on the original data. Furthermore, we used separated networks for both imputation and compression model, in which the calculation cost of the network increased linearly with the number of genome-wide polymorphisms (i.e., the dimension of genomic data), whereas the calculation cost of other popular methods increased with square order, which is also another novel aspect of the proposed method. To the best of our knowledge, there are no prediction methods that can predict the phenotypes of a target trait based on compressed genome-wide polymorphism data in animal and plant breeding.

2. Methodology

2.1 Autoencoder Genome Imputation (AGI)

Autoencoder Genome Imputation (AGI) (Figure 2.1) model can impute missing genome-wide polymorphism data of rice using autoencoder based on the Deep Neural Network. The AGI model demonstrates that by using an autoencoder, it is possible to achieve accurate imputation of missing values in genome-wide polymorphism data. In the AGI model, at first we took the SNP input data and split the input data into several parts. Next, we converted the splitted input data into one hot encoding. We generated several separated networks and trained the separated autoencoder models. After training the separated autoencoder model, we combined the imputed missing genome-wide polymorphisms data.

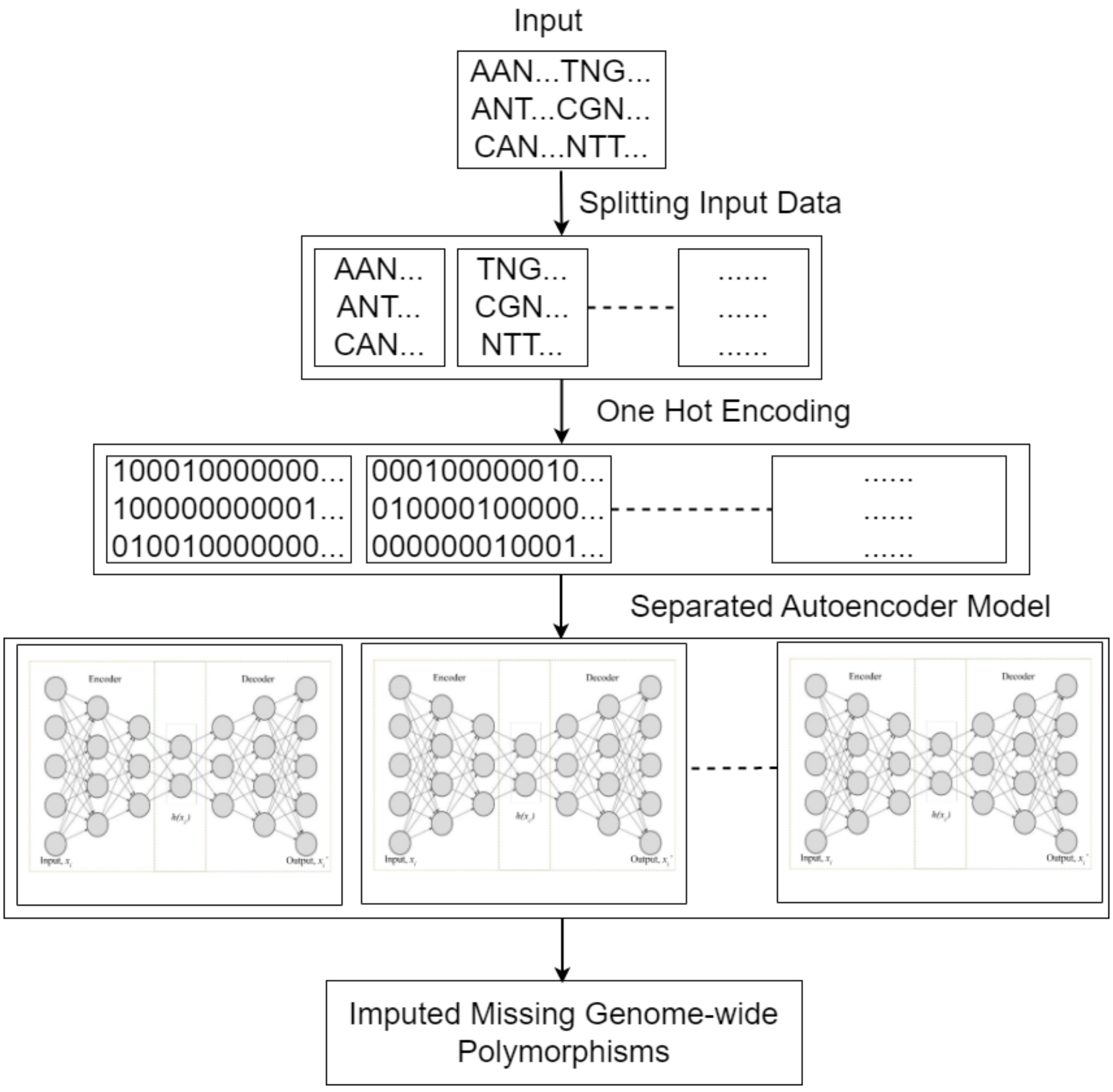


Figure 2.1: Overview of AGI Model.

In the present study, the AGI models were developed by following 3 steps:

Step 1: At first, we imputed the missing values 'N' by '0000'.

Step 2: The separated autoencoder model was trained to optimize mean squared error and to impute the missing genome-wide polymorphism data.

Step 3: After training the separated autoencoder networks, we combined the missing genome-wide polymorphisms and evaluated the imputed value with the true value.

2.1.1 Dataset and Data Pre-processing

To impute missing values we used a genotype dataset which was obtained from a multiparent advanced generation intercross (MAGIC) population derived from eight indica rice varieties (Fedearroz 50, Shan-Huang Zhang-2, IRRI123, IR77186-122-2-2-3, IR77298-14-1-2-10, IR4630-22-2-5-1-3, IR45427-2B-2-2B-1-1, Sambha Mahsuri + Sub1)**.** The dataset comprises genotypes of genome-wide polymorphisms of 1,316 lines for 27,041 SNPs in an excel file.

We pre-processed categorical values (A, C, G, and T) by applying one-hot encoding. In addition, all genomes were encoded into one-hot encoding using a 4-bit coding scheme; that is, $x \in R_{d\times 4}$, where d is the length of the genome sequence. After processing the raw data (1316 x 27041) through One Hot Encoding, the encoded data was transformed into a Python NumPy array. Now the shape of data was (1316 x 108164). As the shape of input data was large, the input data splitting technique was applied to reduce the computational time at a time. By considering 3863 to be the number of splits, numpy hsplit was used to split the one-hot encoded array horizontally (axis =1 i.e. 108,164). Each split contains (1316 x 28) of data, i.e. an input layer with 28 neurons in each network.

2.1.2 AGI Model Training

After splitting input data, each split contains (1316, 28) of data which are divided into 3 subsets: train, test and validation, by first dividing into training and testing sets with 80:20 split considered to be good. The 20% is kept aside as the test until our model is ready for testing. The remaining 80% data is further divided into train and validation with 60:20 split. This means 60% of our data was used for training, 20% was used for validating and 20% for testing. To achieve the optimum performance from our model, there are various parameters that can be tuned to improve performance. We used the 'RandomizedSearchCV' approach for tuning the values from one range above and below the default ranges provided in TensorFlow.

The architecture selected for the AGI model using MAGIC population dataset had three hidden layers in both the encoder and decoder networks (Figure 2.2). The input layer of a network had 28 nodes, the first hidden layer had 14 nodes, the second hidden layer had 7 nodes, with a code size of 3. The network was trained with the Adam optimizer using a learning rate of 0.001. ReLU activation was applied to all layers of the encoder and decoder, except the middle and last layers, for which we applied the sigmoid activation function. The model was trained with MSE loss, and the mini-batch size was 52. The epochs were set to 100 for the AGI model.

2.1.3 Results and Discussion

The presence of missing values is a frequent problem in the analysis of genome-wide polymorphism data. In the original dataset, there was no missing value. We simulated a range of missing proportions at 5%, 10%, 15% and 20% of the original data. The imputed genotypes and true genotypes of the simulated missing entries have been compared to find the accuracy of missing value imputation. Four approaches were compared to validate the imputation accuracy (Table 2.1).

- Imputing missing Genome-wide Polymorphisms by AGI (Replacing 0): The missing value 'N' was replaced by 0, while converting the genome-wide polymorphisms to one-hot encodes. After training the autoencoder model, the decoded genotypes were compared with true genotypes of the simulated missing entries based on their one-hot encodes to calculate the accuracy of the imputation.
- Imputing missing Genome-wide Polymorphisms by Simple Imputer (SI): A simple statistical approach was used to impute missing values. In this approach, the missing value has been replaced by the most frequent value at the polymorphism and then converted the replaced value into a one hot encode.

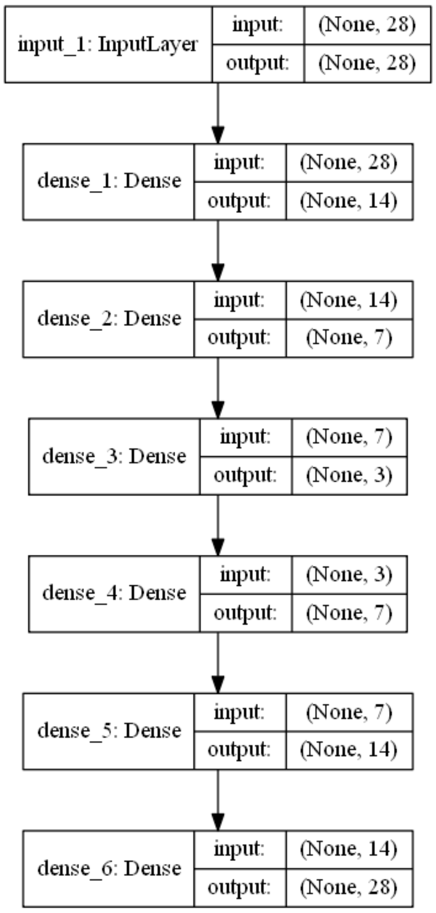


Figure 2.2: AGI Model Architecture.

- Imputing missing Genome-wide Polymorphisms by SI_AGI: The autoencoder model was trained by taking simple imputed values as an input. After training the autoencoder model, the decoded genotypes were compared with true genotypes of the simulated missing entries based on their one hot encode to calculate the accuracy of the imputation.
- Imputing missing Genome-wide Polymorphisms by BEAGLE: A common imputation method using BEAGLE, which is a familiar program in genomic data analysis, was applied. In this method, the missing genome data were imputed by using BEAGLE 5.1 (Browning et al., 2018).

Table 2.1. Summary of missing proportions and the accuracies of imputation methods.

Methods	**Accuracy**			
	5%	10%	15%	20%
AGI (Replacing 0)	94.15	88.37	81.00	72.04
SI	92.48	86.17	80.85	76.44
SI_AGI	95.97	93.25	89.89	86.39
BEAGLE	99.91	99.80	99.69	98.40

Although the accuracy of BEAGLE method was higher than AGI method, BEAGLE does not provide any function for compression of genome-wide polymorphism data. AGI method enables compressing genome-wide polymorphism data and impute missing values in the data at the same time.

2.2 Deep Learning Compression-based Genomic Prediction (DeepCGP)

DeepCGP (Figure 2.3) model can compress genome-wide polymorphism data and use compressed information to predict the phenotype of rice. DeepCGP consists of two models. The first is an autoencoder model that compresses genome-wide polymorphism data. The second is a regression model that takes the compressed information generated by the autoencoder as input and attempts to predict the genotypic values of a target trait. Regression models such as random forests (RF), GBLUP, BayesB, etc., can be used for prediction.

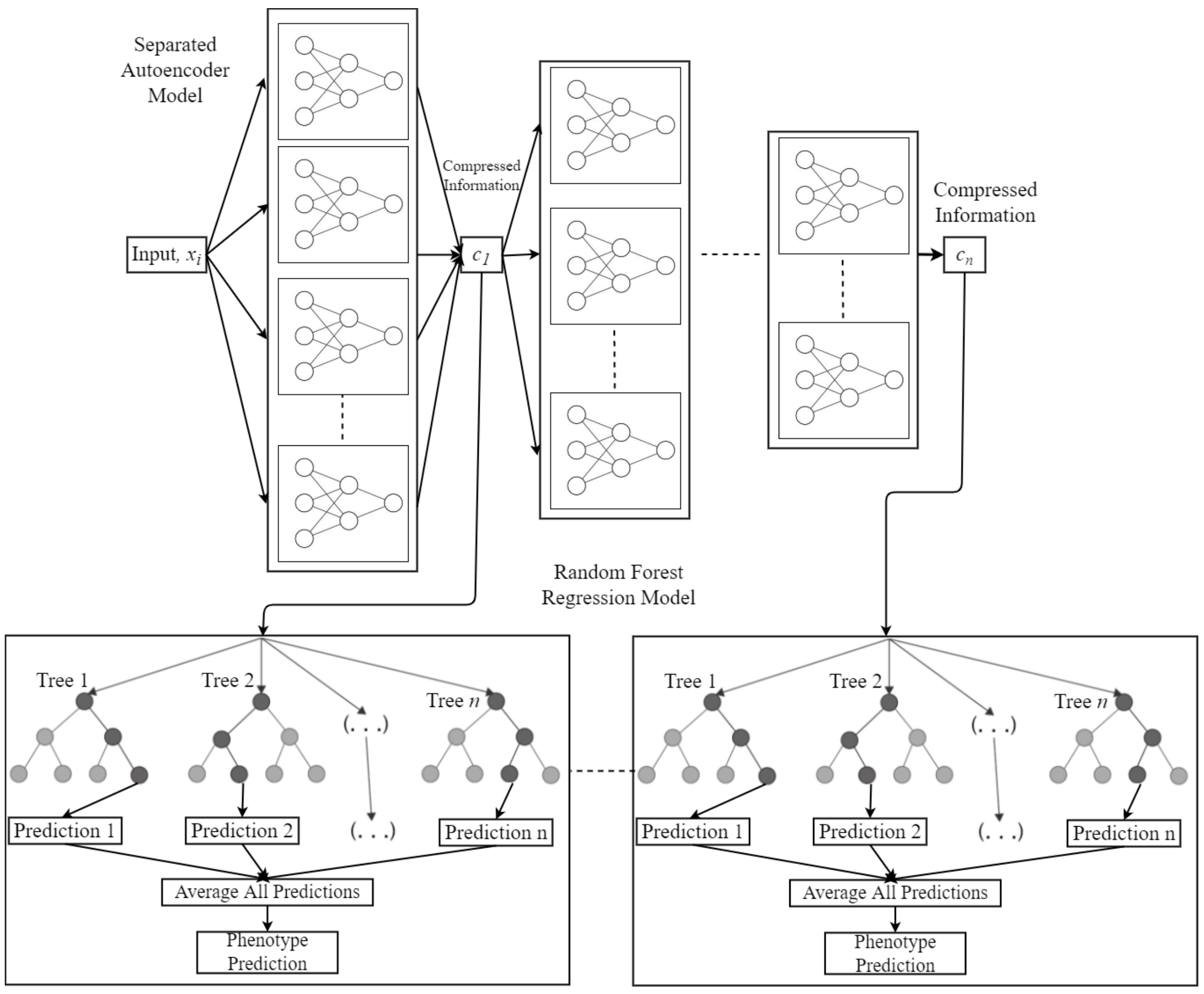


Figure 2.3: DeepCGP architecture. The system consists of autoencoder models and a regression model (i.e. Random Forest [RF]). The autoencoder model compresses the genome-wide polymorphism data, and the regression model (RF) predicts the phenotypes of traits based on compressed genome-wide polymorphism information.

In the first model, our aim was to compress the genome-wide polymorphism data to the maximum limit. To achieve this, we generated several separated networks and trained the separated autoencoder models. The separated autoencoder models compressed the data, which was defined as

Compress_1, c_1. To further compress the genome data, c_1 was used as the input, and the separated autoencoder models were trained for the second compression. The second compression was defined as Compress_2, c_2. In this manner, the separated autoencoder models can compress any genomic data.

After compressing, the regression model will be established to predict phenotypes of a target trait of genotypes (plants/lines). In this study, we used rice germplasm accessions.

The models were trained in three steps:
Step 1: The separated autoencoder model was trained to optimize mean squared error and to compress the genome-wide polymorphism data.
Step 2: The regression model mapped the compressed information to phenotypes of the target traits of rice germplasm accessions.
Step 3: After mapping each compression with phenotypes, we trained a regression model.

2.2.1 Dataset and Data Pre-processing

We used two datasets of different sizes to demonstrate the broad applicability of our model.
C7AIR: The first dataset used was the Cornell-IR LD Rice Array (C7AIR) (Morales *et al.*, 2020), which offers a second-generation SNP array containing 189 rice accessions for 7,098 markers from the Rice Diversity project. The 189 lines had estimated genotypic values for plant height.
HDRA: The second dataset was the high-density rice array (HDRA) (McCouch *et al.*, 2016). The HDRA dataset consisted of 1,568 diverse inbred rice varieties with 700,000 SNPs. Among these lines, the genotypic averages of 34 traits were estimated for 388 lines (Zhao *et al.*, 2011), with some missing records. As 29 genotypes had more than or equal to 10 missing data, while 359 had less than 10 missing data, we chose 359 lines in 18 traits. Furthermore, the genotype dataset was formatted as a bed matrix in vcf format, in which each entry was scored as 0, 1, or 2, where 1 was identified as heterozygous. Since the accessions used in the study were all inbred lines and were expected to be homozygous in most SNPs, we considered 1 as a missing value and converted 0 and 2 to categorical values A (adenine), C (cytosine), G (guanine), and T (thymine). Subsequently, we saved the output in the csv format. Furthermore, we used the 'gaston' package (Perdry and Dandine-Roulland, 2018) in R for this conversion.

We pre-processed categorical values (A, C, G, and T) in the same way as the data pre-processing for missing value imputation by applying one-hot encoding. The C7AIR and HDRA dataset has ~13% and ~10% missing genotypes, respectively. Therefore, we encoded the missing values by "0000." After processing the raw data through one hot encoding, the dimensions of the C7AIR and HDRA data were 189×28392 and 1568×2800000, respectively. As the dimension of the input data was large, an input data splitting technique was applied, which reduced the computational time. We used the NumPy hsplit to split the one-hot encoded array horizontally (axis = 1, i.e., 28,392 and 28,00,000 for C7AIR and HDRA, respectively). For C7AIR and HDRA, each split contained 189×28 and 1568×28 of data, respectively, that is, an input layer with 28 neurons in each network. Moreover, 1,014 and 100,000 separated autoencoder networks were employed for the C7AIR and HDRA datasets, respectively.

2.2.2 Autoencoder Genome Compression (AGC) Model

An autoencoder model was utilized to compress the genome-wide polymorphism data. Each dataset was divided into training (60%), testing (20%), and validation sets (20%) using the scikit-learn 'train_test_split' library. To achieve the optimum performance of a compression model, for both datasets, we executed Keras wrapper class 'KerasRegressor', which permitted us to tune hyperparameters (Table 2.2) using scikit-learn's 'RandomizedSearchCV'. Since the dimension of the C7AIR dataset is low, we tuned the hyperparameters on a whole dataset. For the HDRA dataset, we tuned the hyperparameters on a small subset of training data, i.e. first 1000 splits of data where each split contained (1568 × 28).

Table 2.2. Hyperparameters determined for C7AIR and HDRA datasets.

Hyperparameters	C7AIR		HDRA		
	Compress_1	Compress_2	Compress_1	Compress_2	Compress_3
Optimizer	Adam	Adam	Adam	Adam	Adam
Learning Rate	0.001	0.001	0.001	0.001	0.001
Neurons	28, 14, 7, 3	36, 28, 10, 5	28, 14, 7, 3	30, 15, 5	25, 14, 5
Batch Size	52	32	52	32	32
Epochs	200	200	200	100	150
Loss Function	MSE	MSE	MSE	MSE	MSE

For the C7AIR genotype data, the selected model had three hidden layers in both the encoder and decoder networks. In Compress_1, the input layer of a network has 28 nodes, the first hidden layer has 14 nodes, the second hidden layer has 7 nodes, with a code size of 3. For further compressing the data, the first compression data (c_1) was used as an input in Compress_2. In Compress_2, the input layer of a network has 36 nodes, the first hidden layer has 28 nodes, and the second hidden layer has 10 nodes, with a code size of 5. Both compressions were trained with the Adam optimizer using a learning rate of 0.001. ReLU activation was applied to all layers of the encoder and decoder, except the middle and last layers, for which we applied the sigmoid activation function. The model was trained with MSE loss, and the mini-batch size was 52 for Compress_1 and 32 for Compress_2. The epochs were set to 200 for both the compressions.

The architecture selected for HDRA genotype data was very similar, except for the number of compressions, training epochs, and network structure. The data was compressed to the greatest extent as the HDRA dataset had very high dimensions. The number of nodes in each layer was [28, 14, 7, 3], [30, 15, 5], and [25, 14, 5] for Compress_1, Compress_2, and Compress_3, respectively. Compress_1 was trained with 200 epochs and 52 batch sizes, Compress_2 with 100 epochs and 32 batch sizes, and Compress_3 with 150 epochs and 32 batch sizes. Moreover, the remaining parameters were the same as those for the C7AIR network.

The compression model was implemented using Keras functional API (Ketkar, 2017), which is written in Python and built on top of Tensorflow.

2.2.3 Prediction Model

In the present study, three prediction models, such as RF, GBLUP, and BayesB were used. In addition, we used compressed information as input and extracted the compressed data as a matrix from each dataset. Furthermore, we prepared the estimated genotypic values of a target trait omitting missing entries and arranged them in the same order as the compressed data. A prediction model for each trait was built separately. To evaluate the accuracy of the prediction models and to compare the accuracy among different compression levels, 10-fold cross-validation with five repetitions were applied, and the results were averaged. We used the same folds for all compression levels to ensure that the results were directly comparable. Furthermore, the prediction ability using the correlation coefficient between the estimated and predicted genotypic values was evaluated. Moreover, we evaluated the accuracy of a prediction model based on the original uncompressed genome-wide polymorphism data. Before building the prediction model, we processed the original uncompressed data converting A,T,G,C to 0 and 1 and NA to average values of 0s and 1s, respectively.

A RF model was implemented using the ‘ranger’ R package (Wright and Ziegler, 2017), which is the fastest and most memory-efficient package to analyze high-dimensional data (Breiman, 2001). To train the RF model, we used default parameter settings of the ‘ranger’ function (num.trees: 500, mtry: square root of the number of tuning hyperparameters). To implement GBLUP and BayesB, we used the ‘BGLR’ package (Pérez and de los Campos, 2014) in the R language. The MCMC (Markov Chain Monte Carlo) was run for 25,000 iterations with a 5,000 burn-in period for both GBLUP and BayesB.

All experiments in this study were conducted on a PC with an Intel(R) Core (TM) i9-10980XE, 3.00 GHz CPU, 128 GB RAM, GPU RTX 3090, and a 64-bit Windows 10 Pro operating system.

2.2.4 Result and Discussion

2.2.4.1 Compression of genome-wide polymorphism data

The first experiment in this study was aimed to demonstrate the compression ability of DeepCGP for genome-wide polymorphism data. C7AIR and HDRA datasets were used to train separated stacked autoencoders and evaluate the model by calculating the training time and information loss. Furthermore, the compression ratios were calculated for both datasets; compression ratio is defined as the dimension reduction relative to the uncompressed size, and is given as follows:

$$Compression\,Ratio = 1 - h/x \times 100 \qquad (1)$$

where h is the dimension after compression and x is the dimension before compression. Table 2.3 lists the dimensions of the compressed data, training time, MSE loss, and compression ratio for the C7AIR and HDRA datasets.

Table 2.3 Compression analysis for C7AIR and HDRA datasets.

	C7AIR (189, 7098)		HDRA (1568, 700000)		
	Compress_1	Compress_2	Compress_1	Compress_2	Compress_3
Dimension	(189, 3042)	(189, 425)	(1568, 300000)	(1568, 50000)	(1568, 10000)
Training Time	01h:25m:54s	00h:10m:33s	10d:20h:3m:28s	23h:30m:25s	6h:34m:40s
MSE Loss	0.051	0.039	0.0156	0.0749	0.0911
Compression Ratio	57.14%	94.01%	57.14%	92.86%	98.57%

2.2.4.2 Prediction of phenotypes based on the compressed data

To evaluate the accuracy of the models and to investigate the compressed data, the prediction models were fitted to the compressed data. Figure 2.4a and 2.4b show the prediction accuracy of RF for different compression levels, including non-compression, for both datasets. We considered the compression level according to the compression ratio percentage, which was 0% (original uncompressed data), 57% (57.14%), and 94% (94.01%) for the C7AIR dataset and 0% (original uncompressed data), 57% (57.14%), and 98% (98.57%) for the HDRA dataset.

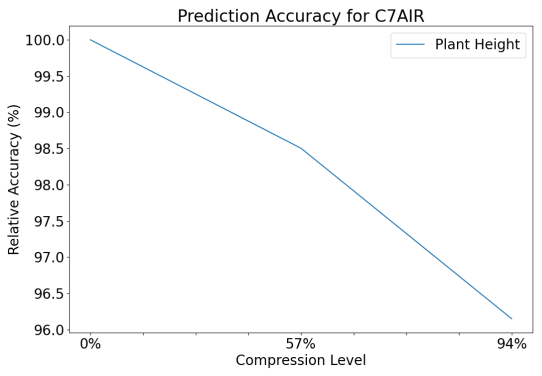
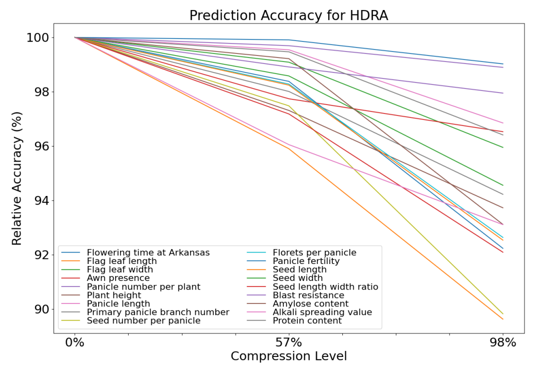
(a)	(b)
	

Figure 2.4: RF prediction relative accuracy (a) C7AIR (b) HDRA datasets.

For the C7AIR dataset (Figure 2.4a) an accuracy similar to that of the original data (with an average difference of approximately less than 3%) was attained even at 94% compression. For the HDRA dataset (Figure 2.4b), accuracies close to the original data were obtained after 98% compression (with an average difference of approximately 5%) for all the selected 18 traits. Moreover, DeepCGP could successfully predict phenotypes even after high-level compression.

The predictive performance was compared between RF and two quantitative genetic models, BayesB (Meuwissen *et al.*, 2001) and GBLUP (VanRaden, 2008). Both models are commonly used in genomic prediction; Figs 2.5a and 2.5b display the predictive performance of BayesB, GBLUP, and RF for the C7AIR and HDRA datasets.

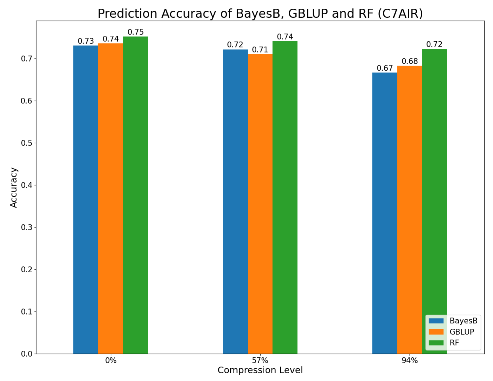
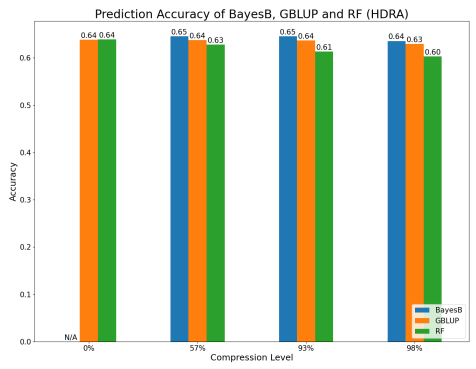
(a)	(b)
	

Figure 2.5: Comparison of BayesB, GBLUP, and RF model prediction accuracies (a) C7AIR (b) HDRA.

We evaluated the predictive performance of the original uncompressed data to the compressed data for both datasets. In the C7AIR dataset, the largest predictive performance was achieved by RF (0.72), followed by GBLUP (0.68) and BayesB (0.67), despite 94% compression. Contrarily, after 98% compression, the largest predictive performance of the HDRA dataset was delivered by BayesB (0.64) followed by GBLUP (0.63) and RF (0.60). The results suggest that RF yielded the highest accuracy of prediction for both original uncompressed data and compressed data of low-dimensional datasets (i.e., C7AIR). In contrast, it is difficult to apply BayesB to a high-dimensional original uncompressed dataset (i.e., HDRA) owing to computational requirements. Therefore, we avoided calculating the prediction accuracy for the original uncompressed HDRA dataset, which is considered as N/A in Figure 2.5b. However, BayesB was applied to compressed HDRA data, and its prediction accuracy outperformed both GBLUP and RF. After 98% compression, the predictive performances of BayesB model were 0.01 and 0.04 higher than that of GBLUP and RF, respectively.

The prediction times for RF, GBLUP, and BayesB were shorter at higher compression levels (Table 2.4). RF demonstrates the lowest time for both datasets at all compression levels compared to the other methods. For the HDRA dataset, BayesB takes a longer time for predicting even after applying compression; BayesB cannot be applied to the original data (i.e., 0%) owing to computational requirements, hence it is considered as N/A.

Table 2.4 Prediction times of RF, BayesB, and GBLUP for C7AIR and HDRA datasets

Methods	C7AIR			HDRA			
	0%	57%	94%	0%	57%	93%	98%
RF	9.26s	5.36s	4.29s	01h:53m: 50s	55m:24s	12m:39s	3m:31s
GBLUP	38.79s	41.1s	37.5s	2h:16m: 37s	2h:13m: 5s	2h:12m: 34s	2h:12m: 21s
BayesB	12m:43s	3m:29s	49.03s	N/A	7d:5h: 1m:31s (Average time per trait)	1d:21h: 39m:43s	08h:57m: 20s

3. Conclusion

The research in this study was inspired after realizing that there are no prediction methods that can predict the phenotypes of a target trait based on compressed genome-wide polymorphism data in animal and plant breeding. Therefore, we wanted to seize the opportunity to investigate if and how state-of-the-art deep learning techniques could be applied to genomic prediction, with a focus on future computer-assisted agriculture. In summary, a novel deep learning model was introduced as a new paradigm to impute and compress genome-wide polymorphism data, which successfully predicts phenotype from the compressed information. The main advantages of our methods are:

- The learning method is scalable to high-dimensional genomic data.
- The proposed method predicts phenotypes of target traits based on compressed genome-wide polymorphism data instead of original (i.e., uncompressed) data.
- Despite the compression of several cycles, the proposed method retains high-quality information, and the prediction accuracy of our method is close to that of genomic prediction based on the original data.
- We used multiple autoencoder networks, in which the computational cost of the network increased linearly with the number of genome-wide polymorphisms (i.e., the dimension of the genomic data), whereas the computational cost of other popular methods increased with the square order, which is also another novel aspect of the proposed method.
- Compression is beneficial for saving both storage and computational time.
- The proposed system improves the performance of data analysis by imputing missing values.

References

Abdella,M. and Marwala,T. (2005) The use of genetic algorithms and neural networks to approximate missing data in database. In, *IEEE 3rd International Conference on Computational Cybernetics, 2005. ICCC 2005.* IEEE, pp. 207–212.

Abdollahi-Arpanahi,R. *et al.* (2020) Deep learning versus parametric and ensemble methods for genomic prediction of complex phenotypes. *Genet. Sel. Evol.*, **52**, 12.

Absardi,Z.N. and Javidan,R. (2019) A Fast Reference-Free Genome Compression Using Deep Neural Networks. In, *2019 Big Data, Knowledge and Control Systems Engineering (BdKCSE)*. IEEE, pp. 1–7.

Beaulieu-Jones,B.K. and Moore,J.H. (2017) Missing data imputation in the electronic health record using deeply learned autoencoders. *Pac. Symp. Biocomput.*, **22**, 207–218.

Bhukya,R. *et al.* (2020) Compression for DNA sequences using huffman encoding. In, Tuba,M. *et al.* (eds), *Information and communication technology for sustainable development: proceedings of ICT4SD 2018*, Advances in intelligent systems and computing. Springer Singapore, Singapore, pp. 615–624.

Blondel,M. *et al.* (2015) A ranking approach to genomic selection. *PLoS ONE*, **10**, e0128570.

Breiman,L. (2001) Random Forests. *Springer Science and Business Media LLC.*

Browning,B.L. and Browning,S.R. (2009) A unified approach to genotype imputation and haplotype-phase inference for large data sets of trios and unrelated individuals. *Am. J. Hum. Genet.*, **84**, 210–223.

Browning,B.L. *et al.* (2018) A One-Penny Imputed Genome from Next-Generation Reference Panels. *Am. J. Hum. Genet.*, **103**, 338–348.

Deutsch,P. (1996) DEFLATE Compressed Data Format Specification version 1.3 RFC Editor.

Duan,Y. *et al.* (2016) An efficient realization of deep learning for traffic data imputation. *Transportation Research Part C: Emerging Technologies*, **72**, 168–181.

Fan,J. *et al.* (2014) Challenges of big data analysis. *Natl Sci Rev*, **1**, 293–314.

Gad,I. *et al.* (2020) A robust deep learning model for missing value imputation in big NCDC dataset. *Iran J. Comput. Sci.*

Gianola,D. *et al.* (2009) Additive genetic variability and the Bayesian alphabet. *Genetics*, **183**, 347–363.

Gianola,D. (2013) Priors in whole-genome regression: the bayesian alphabet returns. *Genetics*, **194**, 573–596.

González-Recio,O. and Forni,S. (2011) Genome-wide prediction of discrete traits using Bayesian regressions and machine learning. *Genet. Sel. Evol.*, **43**, 7.

González-Recio,O. *et al.* (2014) Machine learning methods and predictive ability metrics for genome-wide prediction of complex traits. *Livest. Sci.*, **166**, 217–231.

Goyal,M. *et al.* (2019) Deepzip: lossless data compression using recurrent neural networks. In, *2019 Data Compression Conference (DCC)*. IEEE, pp. 575–575.

Gupta,P.K. *et al.* (2014) Association mapping in crop plants: opportunities and challenges. *Adv. Genet.*, **85**, 109–147.

Habier,D. *et al.* (2011) Extension of the bayesian alphabet for genomic selection. *BMC Bioinformatics*, **12**, 186.

Hamblin,M.T. *et al.* (2011) Population genetics of genomics-based crop improvement methods. *Trends Genet.*, **27**, 98–106.

He,T. and Li,C. (2020) Harness the power of genomic selection and the potential of germplasm in crop breeding for global food security in the era with rapid climate change. *Crop J.*, **8**, 688–700.

Hinton,G.E. and Salakhutdinov,R.R. (2006) Reducing the dimensionality of data with neural networks. *Science*, **313**, 504–507.

Jannink,J.-L. *et al.* (2010) Genomic selection in plant breeding: from theory to practice. *Brief. Funct. Genomics*, **9**, 166–177.

Ketkar,N. (2017) Introduction to Keras. In, *Deep Learning with Python*. Apress, Berkeley, CA, pp. 97–111.

Kole,C. *et al.* (2015) Application of genomics-assisted breeding for generation of climate resilient crops: progress and prospects. *Front. Plant Sci.*, **6**, 563.

Marchini,J. *et al.* (2007) A new multipoint method for genome-wide association studies by imputation of genotypes. *Nat. Genet.*, **39**, 906–913.

McCouch,S.R. *et al.* (2016) Open access resources for genome-wide association mapping in rice. *Nat. Commun.*, **7**, 10532.

Meuwissen,T.H. *et al.* (2001) Prediction of total genetic value using genome-wide dense marker maps. *Genetics*, **157**, 1819–1829.

Morales,K.Y. *et al.* (2020) An improved 7K SNP array, the C7AIR, provides a wealth of validated SNP markers for rice breeding and genetics studies. *PLoS ONE*, **15**, e0232479.

Ogutu,J.O. *et al.* (2011) A comparison of random forests, boosting and support vector machines for genomic selection. *BMC Proc.*, **5 Suppl 3**, S11.

Ogutu,J.O. *et al.* (2012) Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions. *BMC Proc.*, **6 Suppl 2**, S10.

Okser,S. *et al.* (2014) Regularized machine learning in the genetic prediction of complex traits. *PLoS Genet.*, **10**, e1004754.

Perdry,H. and Dandine-Roulland,C. (2018) gaston: Genetic Data Handling (QC, GRM, LD, PCA) & Linear Mixed Models. R package version 1.5.4.

Pérez,P. and de los Campos,G. (2014) Genome-wide regression and prediction with the BGLR statistical package. *Genetics*, **198**, 483–495.

Qaim,M. (2020) Role of new plant breeding technologies for food security and sustainable agricultural development. *Applied Economic Perspectives and Policy*.

Qiu,Y.L. *et al.* (2018) A deep learning framework for imputing missing values in genomic data. *BioRxiv*.

Rana,S. *et al.* (2012) Robust regression imputation for analyzing missing data. In, *2012 International Conference on Statistics in Science, Business and Engineering (ICSSBE)*. IEEE, pp. 1–4.

Scheet,P. and Stephens,M. (2006) A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase. *Am. J. Hum. Genet.*, **78**, 629–644.

Schmidhuber,J. (2015) Deep learning in neural networks: an overview. *Neural Netw.*, **61**, 85–117.

Selby,P. *et al.* (2019) BrAPI-an application programming interface for plant breeding applications. *Bioinformatics*, **35**, 4147–4155.

Silva,M. *et al.* (2020) Efficient DNA sequence compression with neural networks. *Gigascience*, **9**.

Stephens,Z.D. *et al.* (2015) Big data: astronomical or genomical? *PLoS Biol.*, **13**, e1002195.

Swaminathan,R. *et al.* (2016) A review on genomics apis. *Comput. Struct. Biotechnol. J.*, **14**, 8–15.

Szymczak,S. *et al.* (2009) Machine learning in genome-wide association studies. *Genet. Epidemiol.*, **33 Suppl 1**, S51-7.

Tester,M. and Langridge,P. (2010) Breeding technologies to increase crop production in a changing world. *Science*, **327**, 818–822.

Thudi,M. *et al.* (2021) Genomic resources in plant breeding for sustainable agriculture. *J. Plant Physiol.*, **257**, 153351.

VanRaden,P.M. (2008) Efficient methods to compute genomic predictions. *J. Dairy Sci.*, **91**, 4414–4423.

Wang,R. *et al.* (2018) DeepDNA: a hybrid convolutional and recurrent neural network for compressing human mitochondrial genomes. In, *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, pp. 270–274.

Wang,R. *et al.* (2019) Human mitochondrial genome compression using machine learning techniques. *Hum Genomics*, **13**, 49.

Wright,M.N. and Ziegler,A. (2017) ranger : A Fast Implementation of Random Forests for High Dimensional Data in*C++* and*R*. *J. Stat. Softw.*, **77**, 1–17.

Zhao,K. *et al.* (2011) Genome-wide association mapping reveals a rich genetic architecture of complex traits in Oryza sativa. *Nat. Commun.*, **2**, 467.