A Large Scale Study on Shared Weight Autoencoder

Zhengyang Liu
Research School of Computer Science
The Australian National University
u6022937@anu.edu.au

Abstract. With the development of the deep learning, the application of the unsupervised learning has become more widespread, especially a feature extraction method called autoencoder. However, when training large datasets, besides the training accuracy or the loss, the executing time become an important attribute to judge the model. Two autoencoders with shared weight which can reduce executing time is designed can compared with other methods, one is the traditional artificial neural network(ANN) for a small dataset and the other one is a convolutional neural network(CNN) of a object recognition dataset called CIFAR10. What's more, a regression model is build for the previous dataset to predict the final score of the students in Portuguese class while a classification model is trained to recognize the object label in a image for the CIFAR10. By comparing the results for several experiments, it can be found that the shared weight autoencoder can make the performances of complicate dataset slightly worse, but it can save time and memory when training while the results on the small dataset is quite stable and well.

Keywords: Neural Network, Convolutional Neural Network, Regression, Classification, Shared Weight Autoencoder.

1. Introduction

With the improvement of technology, there are more and more problems that can be solved by computers instead of doing a large amount of calculations manually. A specific problem is about score prediction in school which benefits both students and educational resources managers. Educational level is a vital factor to judge the level of a country. In the past decades, the government of Portugal try a lot to improve the educational level of students, however, the data show that the student failure rate is quite high among the Europe countries as well as the dropping out rates. Because Portuguese course consist of a large amount of fundamental parts which support other courses, it is meaningful to analyze the data that related to the Portuguese grade. Another kind of problem is to recognize objects and label them, which is quite often in our daily life. When there are only few objects or labels, it is easy to recognize them by only observation. Nevertheless, when there tens of thousands of them, visual observation become impossible. How to recognize them using machine become hot topics.

To solve all the complicated problems in a easy way, ANN is created to do both regression and classification task for small dataset training. However, when it comes to large datasets like image datasets, lacking complicated design, NNs performs terribly. Therefore, the deep learning technique is created to deal with the large dataset, including CNN, a feature extraction network proposed by Lecun[2]. The different between CNN and ANN is the convolutional layers, which can extract the features. In general, there are multiple convolution kernels which can do convolution operation on channels of the images. The whole CNN network contains multiple convolution layers, pooling layers and some fully connected layers. To prevent the overfitting situation and increase robustness of the designed model, a autoencoder which is a kind of compression and dimension reduction algorithm is implemented. Two autoencoders are designed in this work, one is a traditional ANN designed to extract the main attributes in the student performance dataset while the other one is a CNN for feature learning and dimensionality reduction. Based on the encoder-decoder method paradigm[3], the input of the autoencoder is first transformed into a typically lower-dimensional space(encoder), and then expended to reconstruct the initial data(decoder). Moreover, when evaluate a model, executing time is also a factor that is of importance. Inspired by the shared weight autoencoder proposed by Tom[1], shared weight method is added to both of the autoencoders to reduce the executing time.

For the score prediction task, a customized regression model is trained given the first and second period grade, developed with a shared weight autoencoder to compress the preprocessed attribute data and extract the valuable features. In addition, since this is a global issue, several studies are addressed on this theme. In 2006, Pardos and his partners predicted math grade using Bayesian Network and their model can achieve 85% accuracy rate[5]. For the object recognition task, the classification model is chosen and developed with a shared weight CNN autoencoder. With the hierarchical designed of stack autoencoders, the accuracy of CIFAR10 dataset can reach 78.2% in Masci's search[4]

In the study of prediction, using transcripts and questionnaires, the data was collected from two secondary schools in Portugal. After preprocessed, the dataset is fed in a shared weight autoencoder net to compress the features.Then a regression model is trained to create a predictor for every input data. The results show that a shared weight autoencoder has a positive influence to make stable prediction, however, the shared weight technique basically dod not work in the small dataset. As for the classification work, the data was from CIFAR10 dataset and after the feature extraction in the autoencoder, the data is fed into a classification NN to give the image labels. The results show illustrate that shared weight autoencoder can performs slight worse , but it can save time when training

The remainder of this paper is organized as follows. Next section goes through the data source and description, followed by some methodologies used in this work. Before demonstrating the results of the work, the computing environment is introduced. Afterward, the final parts draw a conclusion and give the feature work on the whole process.

2 Data source and description

2.1 Input data of student performance dataset

There are two datasets for the student performance in UCI Machine Learning Repository, Mathematics and Portuguese features and scores. Since the number of instances in the Mathematics one is too small (only 398), the Portuguese one is chosen. The data is aggregated after collecting student-related information(e.g. sex, age, mother's education) as well as asking for the school to provide the number of absences and Portuguese course' grade in first, second and final period. In addition, the sore ranks from 0 to 20, where 0 is the lowest and 20 is the best.

2.2 Preprocessing of students performance dataset

The data provided by UCI is the complete one that discards some features that lack discriminative attributes. However, when it is fed into the neural network, the categorical data must be transferred to numerical data. Therefore, one-hot encoding is used to solve this problem, with 33 features changing to 59. Like 'sex' in original dataset, it can be changed to 'Male' and 'Female' attribute with only 0 or 1 in the to represent the sex of a student. What's more, because the range of the input is quite wide, so a normalize method is applied to change the input to a number between 0 to 1 with the ratio between the input maintaining unchanged.

2.3 Final students performance dataset

After one-hot encoding of the raw dataset, the final dataset has 59 numeric attributes related to students' status and 649 instances. The features are that may have influence on the final score of the Portuguese class and are different between students, like demographic situation (e.g. sex, family size), emotional information (e.g. alcohol drinking amount) and student performances in school (e.g. absences, score in every period). The student-related attributes in final dataset is described in appendix and the desired score to predict is the final score (G3).

2.4 CIFAR10 dataset and preprocessing

The Cifar10 dataset consists of 60000 natural color images of 32 × 32 in 10 classes. Each class contains 5000 instances in the training set and 1000 in the testset. Although the images greatly vary inside each class, they are not necessarily centered and may only has part of the object with varying background, which makes the whole dataset very hard to train and do the classification task. Because the same reason mentioned above, the input is normalized to range[-1,1] without change the ratio between any two of the input.

3 Methodology

3.1 ReLU and Sigmoid activation function

The activation functions are extremely important feature in defining a neural network, deciding whether a neuron will be activated or not. If the information that a neuron received is relevant to the target, the neuron is activated follow the formula:

$$Y = F(\sum(w \times I) + b) \tag{1}$$

In above formula, F is the activation function, w is weight, I means input and b refer to bias. Among all the activation functions, the ReLU and Sigmoid function is used in this work. Figure1 shows the shape and formula of both functions.

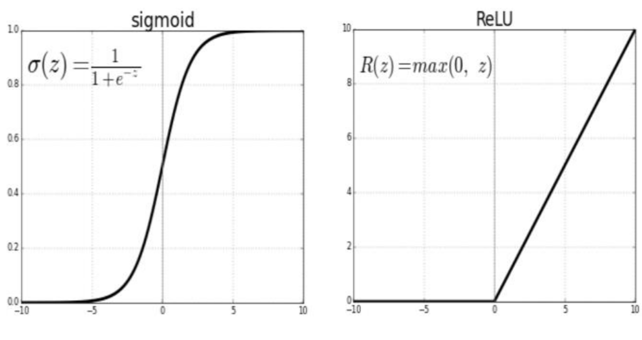

Figure1.Shape and formula of Sigmoid and ReLU

It can be found that Sigmoid function ranges from 0 to 1 while the ReLU function's range is decided by the input. Furthermore, sigmoid will not blow up activations but gradient may get vanish during the computation while the ReLU can solve such problem. Therefore, when the range of the data is considered, choosing the suitable activation is of great importance, like the ReLU is applied in this task for both autoencoder network and regression model, nevertheless, the sigmoid function is selected to constrain the output of the autoencoder net due to the input range.

3.2 Adaptive Moment Estimation (Adam) optimizer

In this work, Adam optimizer is used for training both the autoencoder net and the regression net. Compared with Stochastic Gradient Descent(SGD) optimizer which update the parameter for each training epochs to get the global minima but it leads to a complicated convergence of the accurate minimum and it can keep overshooting, Adam do a better job of converging faster and more exact. Adam stores not only an exponentially decaying average of past squared gradients v(t), but also an exponentially decaying average of past gradients M(t).

$$\widehat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{2}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{3}$$

Therefore, the final update of parameter is as follow:

$$\theta_{t+1} = \theta_t - \frac{n}{\epsilon + \sqrt{\hat{v}_t}} \widehat{m}_t \tag{4}$$

In addition, by rectifying problems that occur in other optimizers like learning rate vanishment and fluctuation of the loss function, the Adam performs well in real work.

3.3 Max-pooling

For hierarchical neural networks in general, especially CNN, translation-invariant representations are obtained by introducing a max-polling layer[1], which down-samples the latent representation by taking the maximum value over nonoverlapping sub-regions. This kind of method improves filter selectivity because the match between the feature and the input field over the region of interest can determine the activation of the neurons in the latent representation.

The max-pooling layers are introduced in the autoencoder net for CIFAR10 to sparsity over the hidden representation by deleting all the values that are smaller than the maximal value in the nonoverlapping area. By adding these layers, the feature detector performs more broadly applicable. In addition, this sparse latent method decreases the average number of filters contributing to the pixels' decoding during the reconstruction stage, which makes filters to be more general.

3.4 Shared weight Autoencoder net

A shared weight autoencoder net is an NN used to learn a representation for a dataset to reducing the dimensionality and the space of weight configurations in unsupervised learning. In these two works, two autoencoder net (compression net) with a weight between the input layer and the hidden layer being equal to that between the hidden layer and the output layer is designed to extract the main features that can represent the whole input set. By doing this, the features are decreased to 25 from 59 before being fed to the regression model in the student performance dataset while the image size decreases from 32×32×3 to 16×8×8. Two autoencoders are described in next two sections

3.4.1 Traditional ANN shared weight autoencoder for the student performance dataset

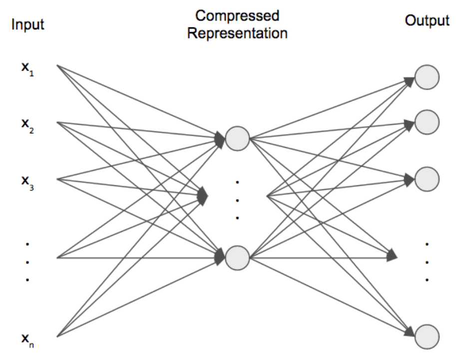

Figure2. Shared weight ANN autoencoder construction

The net is a two-layer feed-forward net with one hidden layer, whose all connections are from one layer to the next one with no other kind of connections like multilayer, lateral or backward connections. Unlike the hidden layer whose size is smaller than that of the input layer, the output layer contains the same neurons as input layer. The hidden layer is designed to compress the features while the output layer can recover them. The net can be thought as two separate things: an encoder and decoder. To achieve weight shared, the weight of encoder and decoder must be equal, using the two formula below:

$$\mathbf{z} = \sigma(\mathbf{W}^{(e)}\mathbf{x} + \mathbf{b}^{(e)}) \tag{5}$$

$$\hat{\mathbf{x}} = \sigma(\mathbf{W}^{(d)}\mathbf{z} + \mathbf{b}^{(d)}) \tag{6}$$

Where the superscript corresponds with encoder and decoder, with x being the input and σrepresenting the activation function. As what just described, W(e) is equal to W(b). Therefore, the decoder is the actual output of the model and the original input is the desired output. Since the input and output data is numeric and it is easy to get the mean, the Mean Square Error(MSE) selected to be the loss function and the formula is:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2. \tag{7}$$

As for optimizer, the Adam described above is used to achieve a fast and exact net. The net is trained using the features of training set which randomly chooses 80% of the whole one with the input data being also the desired output. Moreover, the test set is the remaining 20% of the whole dataset. As for measuring error, back-propagation which is accurate and easy to implement is applied. When defining the activation function of the hidden layer, instead of the sigmoid function that is very popular for its convenience to calculate things like gradient, Relu is applied to reduce the likelihood of gradient vanishing. Because the input is between 0 to 1, it is better to constrain the output to be in the same range. Therefore, a sigmoid function before getting the decoder is added. After training, the encoder that is 25 attributes with 649 instances is formed to be fed into next stage.

3.4.2 CNN shared weight autoencoder for the CIFAR10 dataset

To implement the image compression and reconstruction, a hierarchical CNN shared weight autoencoder is trained. An encoder to do the image feature extraction and a decoder to achieve image reconstruction. The whole autoencoder is designed as shown in Figure3.

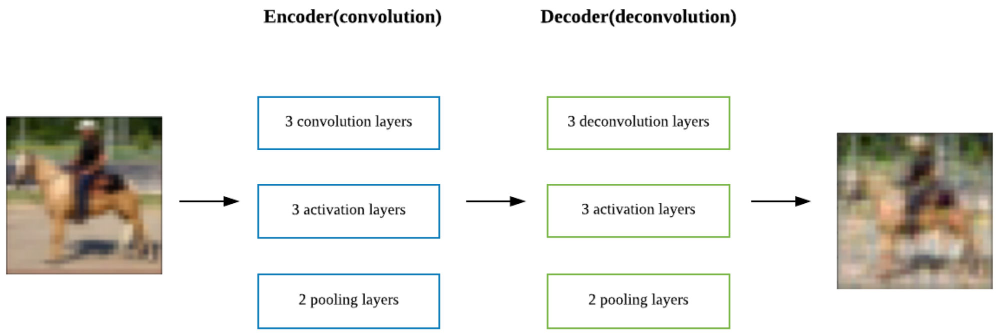

Figure3. Shared weight CNN autoencoder construction

An encoder takes the original image as input and has three convolution layers, three activation layers and two poling layers while the decoder takes encoder as input, having three deconvolution layers, three activation layers as well as two poling layers. To save the memory and reduce executing time when training, the weights of three deconvolution layers are fixed and equal to the corresponding encoder's convolution layers. ReLU activation function is used in activation layers followed by the dropout to increase the robustness of the model because it makes all layers to delete redundant representations before the dropout. When training, the MSE is used because it can be easily calculated and it can well represent the difference between target and the predicted images. Furthermore, for training, back propagation and Adam optimizer are chosen for the same reason discussed above.

3.4 Regression net for student performance dataset

The regression net is designed to be a feed-forward three processing layers network with the input being the encoder which is gained from the traditional ANN shared weight autoencoder. Because the aim of the task is to forecast the final grade of a student, so the output of the regression model consists only one neuron like Figure4.

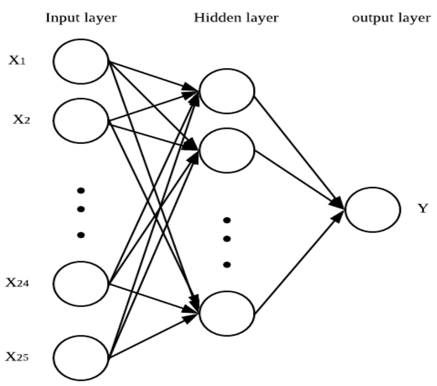

Figure4. Regression net construction

The input is only 25 features so the hidden neurons is a little fewer than input neuron being 10. Since the input dataset size is small, the hidden layer is useless to contain more neuron or be multilayer which will increase the space to record the hidden neuron information and slow down the speed of processing. Relu is also applied in this net because of the same reason as above. Since the score of a course is numeric and different from 0 to 20, the loss function is also the MSE. Adam is used again to develop the regression model. As for test set, the rest of 20% dataset is select.

3.5 Classification net for CIFAR10 dataset

The Classification net is a two-layer feed-forward network whose input is the encoder of the CNN shared weight autoencoder. Since it is a classification task and there are 10 classes in the dataset, the output neurons are 10 like Figure 5.

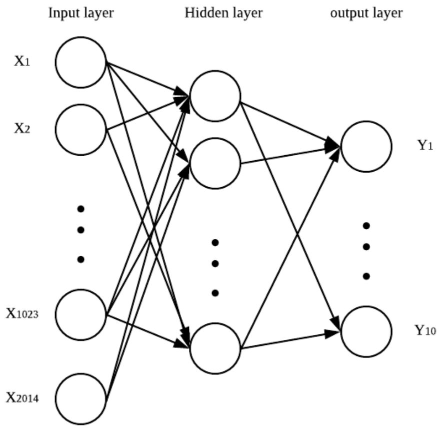

Figure5. Classification net construction

The input is the encoder which is the size of 16×8×8 for each instance and the output is the classes. Since the features have already been reduced from 32×32×3 to the input size which is not the grid-like input, only a traditional NN is designed to do the classification task. Since there are a lot of attributes and if they are fully connected, it needs a large amount of memory and training time will be quite slow, with potential overfitting problem. So some dropouts are added between the input layer and the hidden layer as well as the hidden layer and the output layer. ReLU is applied while the loss function is the CrossEntropy Loss, which measures the performance of a classification model whose output is a probability value between 0 and 1. The separate losses for each class label per observation are calculated and summed up as the formula:

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) \tag{8}$$

3.6 Evaluation methods

For the regression model, it is used to predict a score and is easy to get the desired score from the dataset and the actual output (predicted one) from the regression model, an evaluation method called Root Mean Squared Error(RMSE) is introduced. The result can be computed using the following equation

$$RMSE = \sqrt{\sum_{i=1}^{N} (y_i - \widehat{y}_i)^2 / N} \tag{9}$$

Where the $\widehat{y_i}$ is the actual value of the i-th epochs. It is differentiable so it is not complicated to calculate.

For the classification task, the obvious way to judge a classification model is to compare its predicted label with the desired label and find the accuracy of the prediction. So an accuracy formula is used to test the model

$$accuarcy\_rate = \frac{correct\_predicts}{test\_dataset\_size} \tag{10}$$

4. Computational Environment

All the experiments in this study are conducted using PyTorch, an open source machine learning library for Python and used for applications such as natural language processing. Python is a free and interpreted high-level programming language with loads of powerful tools to analyze data. The PyTorch library presents lots of coherent functions for regression tasks. Particularly, it contains main process functions for training a regression model such as loss functions, optimizers and activation functions. Moreover, all experiments are done on an i5-7500 CPU, Nvidia GTX1050Ti, 8g RAM and windows environment.

5. Results and Discussion

5.1 Result about using traditional NN shared weight autoencoder on the regression task

After completing the NN autoencoder the whole regression model performs slightly better than one that only contains a regression model. However, when apply the shared weight method, the performance is nearly equal to the autoencoder without shared weight.
The Table3 shows the data of both model.

Table3. Test evaluation of the whole model with and without the autoencoder

	Whole model test evaluation
Model without autoencoder	2.3 – 3.4
Model with autoencoder	2.50 ± 0.2
Model with shared weight autoencoder	2.60 ± 0.2

It can be seen that the autoencoder does work. After feeding the input(59 attributes) to compression net, a RMSE of around 0.14 can be achieve to decrease the features to 25 in the test phase. And the evaluation function is the RMSE. It can be concluded that after doing autoencoder, the RMSE of test set is keeping around 2.6 while the result using only regression model without shared weight autoencoder fluctuates between 2.3 to 3.4. Thus, the regression model with an autoencoder can forecast more accurate and stable. Furthermore, the MSE of 500 epochs in both model during the training is shown in Figure4. Due to the little difference among every train time, the selected conditions occur the most times, which illustrates that the regression network is strongly influenced by the input size at the beginning because when the point with x-axis set to 0 the y value is of big difference (140 and 160).

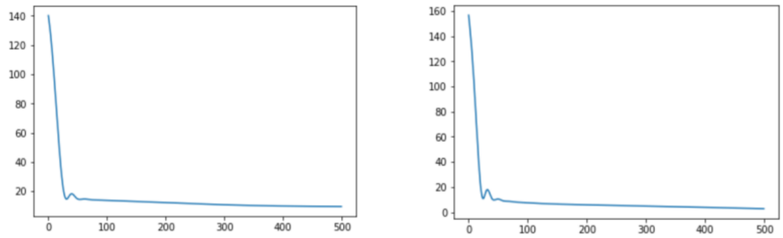

Figure6. The MSE of 500 epochs for regression with(left) and without(right) shared weight autoencoder respectively

As for the share weight method, since it is only apply on the autoencoder, so the training time of the NN autoencoders with and without shared weight is recorded respectively for further comparison. However, because the dataset is quite small, the training time is very short, which has a distinct change during training. Both executing time of the autoencoders are between 0.8-1.1 seconds generally, which are the same. Therefore, an shared weight autoencoder can do a stable prediction job on the small dataset and the shared weight method do not play an important role in speeding up the training process.

5.2 Result about using CNN shared weight autoencoder on the classification task

Unlike training using a small dataset, shared weight method performs well in the larger dataset, with largely decreasing the training time, the accuracy of the whole model only has a minor drop. Figure7 shows the reconstructed images (original input in the left and the reconstructed images on the right) in the beginning of the autoencoder training and the end of the 100 epochs training, with the other information collected for share weight autoencoder in the table4.

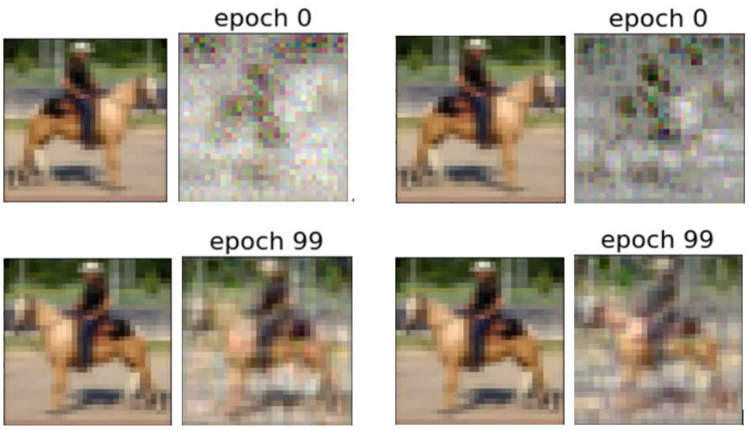

Figure7. Reconstructed images using autoencoder with (4 right images) and without(4 left ones) shared weight

Table4. Vital information using autoencoder with and without shared weight respectively

	Testset loss on the autoencoder	Executing time on the autoencoder (s)	Accuracy on the whole task
Autoencoder without shared weight	0.00022	2145	66.32%
Autoencoder with shared weight	0.00042	1598	64.97%

It can be seen from the figure and table that the shared weight method can make autoencoder loss bigger and make the reconstructed images more blurry, however, the performance of the whole classification model do not decrease much while it improve nearly 25% of performance of the training time. Moreover, the big loss of the autoencoder affect the accuracy slightly. Therefore, the shared weight method is more suitable to apply on the model when training the large dataset.

5.3 Result about comparing CNN shared weight autoencoder and the Stacked Convolutional Auto-Encoders(CAES)(related paper)

Accuracy can reach 64.97% using CNN shared weight autoencoder while the error rate is only 21.8% when applying CAES[4]. CAES is a deep hierarchy formed by stacking several normal autoencoders and each layers' input is from the latent representation of the layer below, which is a greedy, layer-wise fashion way to pre-train the unsupervised model. In addition, the author use CNN which is initialized by the hierarchical trained CAES weights to do the classification task while a traditional ANN is designed to classify the objects in my work, therefore the accuracy of their work is much hair than mine. To improve my work, instead of the traditional ANN, more complicated structure can be used to do classification like CNN because after the autoencoder net, the image size is a little big for a ANN to train.

6 Conclusion and Feature work

To make prediction on the final score of the Portuguese class, a regression model based on neural network is developed with an shared weight autoencoder to extract the valuable features before regression. The model performs well compared with the one that has no autoencoder, nevertheless, because the dataset consists a small number of attributes and the executing time is quite short, the shared weight method do not work. However, when apply a shared weight CNN autoencoder on a large image dataset, it does perform well with significant improvement of the executing time and minor decrease of the accuracy when classify objects

The conclusion can be drawn from the experiments :

a. Because the input size affect a lot on the NN regression model, a autoencoder is developed to extract valuable features and reduce the dimensionality of the input dataset
b. Making the autoencoder weight between the input and hidden neurons equal to that between hidden and output neurons(in CNN is the weight between deconvolution layers and corresponding convolution layers) can lead to improvement in speed when a big dataset needs to be trained and no much decrease of the final result.
c. To simplify the convergence during the training stage, Adam is applied.

Many of the experiments can be done further. First of all, there are many others datasets that can be applied to a regression or classification task. The models in this work should be tested on other datasets to ensure that these two shared weight autoencoder can extract the features of those data as well. Secondly, the classification net can be changed to a more complicated structure like CNN to classify more precise. Finally, some hierarchical structures can be added to the autoencoder to reconstruct a more similar image.

References

[1] Gedeon T. D., Catalan J. A., J. Jin, "Image Compression using Shared Weights and Bidirectional Networks", *Proc. 2nd International ICSC Symposium on Soft Computing (SOCO'97)*, pp. 374-381, 1997.

[2] LeCun, Y., & Bengio, Y. (1995). Convolutional networks for images, speech, and time series. The handbook brain theory and neural networks, 3361(10), 1995.

[3] Ranzato, M., Fu Jie Huang, Y.L.B., LeCun, Y.: Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: Proc. of Computer Vision and Pattern Recognition Conference (2007)

[4] Masci J., Meier U., Cireşan D., Schmidhuber J. (2011) Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction. In: Honkela T., Duch W., Girolami M., Kaski S. (eds) Artificial Neural Networks and Machine Learning – ICANN 2011. ICANN 2011. Lecture Notes in Computer Science, vol 6791. Springer, Berlin, Heidelberg

[5] Cortez P. and Silva A. Using Data Mining to Predict Secondary School Student Performance. In A. Brito and J. Teixeira Eds., Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008) pp. 5-12, Porto, Portugal, April, 2008, EUROSIS, ISBN 978-9077381-39-7.

Appendix

Table4. the student-related attributes after preprocessing

Attribute	Description (domain)
GP	student is from Gabriel Pereira school (numeric: 1 yes, 0 no)
MS	student is from Mousinho da Silveira school (numeric: 1 yes, 0 no)
Male	student is a male (numeric: 1 yes, 0 no)
Female	student is a female (numeric: 1 yes, 0 no)
Urban	student’s home is in urban area (numeric: 1 yes, 0 no)
Rural	student’s home is in urban area (numeric: 1 yes, 0 no)
famsize_GT3	family size is greater than 3 (numeric: 1 yes, 0 no)
famsize_LE3	family size is greater than 3 (numeric: 1 yes, 0 no)
Pstatus_A	parents are living apart (numeric: 1 yes, 0 no)
Pstatus_T	parents are living together (numeric: 1 yes, 0 no)
M_at_home	mother has no job and is at home (numeric: 1 yes, 0 no)
Mh_ealth	mother is engaged in care-related work (numeric: 1 yes, 0 no)
M_other	mother is engaged in other work (numeric: 1 yes, 0 no)
M_services	mother is engaged in services-related work (numeric: 1 yes, 0 no)
M_teacher	mother is engaged in educational-related work (numeric: 1 yes, 0 no)
F_at_home	father has no job and is at home (numeric: 1 yes, 0 no)
F_health	father is engaged in care-related work (numeric: 1 yes, 0 no)
F_other	father is engaged in other work (numeric: 1 yes, 0 no)
F_services	father is engaged in services-related work (numeric: 1 yes, 0 no)
F_teacher	father is engaged in educational-related work (numeric: 1 yes, 0 no)
reason_course	course is the reason why student choose the school (numeric: 1 yes, 0 no)

reason_home	closing to home is the reason why student choose the school (numeric: 1 yes, 0 no)
reason_reputation	school reputation is the reason why student choose the school (numeric: 1 yes, 0 no)
guardian_father	father is the guardian of the student (numeric: 1 yes, 0 no)
guardian_mother	mother is the guardian of the student (numeric: 1 yes, 0 no)
guardian_other	neither mother nor father is the guardian of the student (numeric: 1 yes, 0 no)
schoolsup_no	student does not have extra educational school support (numeric: 1 yes, 0 no)
schoolsup_yes	student has extra educational school support (numeric: 1 yes, 0 no)
famsup_no	student does not have family educational support (numeric: 1 yes, 0 no)
famsup_yes	student has family educational support (numeric: 1 yes, 0 no)
paid_no	student does not attend extra paid classes (numeric: 1 yes, 0 no)
paid_yes	student attends extra paid classes (numeric: 1 yes, 0 no)
activities_no	student does not participate in extra-curricular activities (numeric: 1 yes, 0 no)
activities_yes	student participates in extra-curricular activities (numeric: 1 yes, 0 no)
nursery_no	student have not attended nursery school (numeric: 1 yes, 0 no)
nursery_yes	student has attended nursery school (numeric: 1 yes, 0 no)
higher_no	student does not want to take higher education (numeric: 1 yes, 0 no)
higher_yes	student wants to take higher education (numeric: 1 yes, 0 no)
internet_no	student has no internet access at home (numeric: 1 yes, 0 no)
internet_yes	student has internet access at home (numeric: 1 yes, 0 no)
romantic_no	student is not in a romantic relationship (numeric: 1 yes, 0 no)
romantic_yes	student is in a romantic relationship (numeric: 1 yes, 0 no)
age	student’s age (numeric: from 15 to 22)
Medu	mother’s education level (numeric: from 0 (primary) to 4 (high))
Fedu	father’s education level (numeric: from 0 (primary) to 4 (high))
Traveltime	travel time from home to school (numeric: 1. < 15 min., 2. 15 to 30 min., 3. 30 min. to 1 hour or 4.> 1 hour).
Studytime	weekly study time (numeric: 1. < 2 hours, 2. 2 to 5 hours, 3. 5 to 10 hours or 4. >10 hours)
	failures number of previous courses (numeric: n if $1 \le n < 3$, else 4)
Failures	family relationships’ quality (numeric: from 1 (bad) to 5 (very good))
Famrel	free time level after school (numeric: from 1 (low) to 5 (high))
Freetime	frequency of hanging out with friends (numeric: from 1(quite low) to 5 (very high))
goout	alcohol drinking amount in weekdays (numeric: from 1(quite low) to 5 (very high))
Dalc	alcohol drinking amount in weekend (numeric: from 1(quite low) to 5 (very high))
Walc	health condition (numeric: from 1(quite low) to 5 (very high))
health	
absences	student’s absence in school (numeric: from 0 to 93)
G1	score in first period (numeric: from 0 to 20)
G2	score in second period (numeric: from 0 to 20)
G3	final score (numeric: from 0 to 20)