Breast Cancer Analysis Using Neural Network Model and Shared Weights Autoencoder

Author Name (Paper 195 assignment 1)

Research School of Computer Science, Australian National University
u1234567@anu.edu.au

Abstract. As the base for breast cancer analysis, cell information such as cell size, shape and texture features has been widely used. In the past, people used traditional machine learning methods to analyze the cell information and classify the cells from benign and malignant. In this paper, I use neural network method to implement the breast cell classification task. To solve the redundant feature problem, I add autoencoder in my model feature extraction. Then to improve the feature extraction ability, I use shared weights technique in the autoencoder. I conduct several experiments with the Breast Cancer Wisconsin(Diagnostic) dataset that has well collected data. My experiments investigate the impact of autoencoder and the effectiveness of shared weights technique. Furthermore, I compare the performance of the model that uses shared weights autoencoder with logistic regression. My results confirm three conclusions: the autoencoder has a good feature extraction ability. The shared weights technique can aid autoencoder extract more important feature. And my model shows a better performance than logistic regression.

Keywords: Neural network, Shared weights, Autoencoder, Breast cancer, Compressed feature

1 Introduction

Cancer prediction is an approach of using different bio-information to predict whether a cell is malignant. To improve the prediction accuracy, people use the computer to collect the cell features. With the aid of computer, microscope image can be converted to accurate feature information, it can highly improve prediction accuracy and objective feature assessment [6]. And the cancer prediction is an essential part in cancer treatment and cancer prevent.

Cancer prediction with cell information can be regarded as a simple classification task in machine learning area. In general, a classification task is using some features of an object to tell which category it belongs to. There are many methods to implement this task. For instance, logistic regression [1], C4.5 decision trees [4] and Naïve Bayesian learning (NB) [5] are often be used for the classification task.

In the past, people have tried different method on this cancer prediction task. For example, W. H. Wolberg et al uses logistic regression and inductive machine learning method, these two methods provide the base for breast cell analysis [1]. G. R. M. A. Sizilio et al uses fuzzy system, this method can reduce the variation hit of malignant cases [7]. Y. M. George et al test the performance of SVM, this method can create a soft margin that permits some misclassification [8].

Traditional machine learning methods have shown a good performance in the breast cancer classification task, for example, logistic regression method gives the accuracy value of 96.2% [1]. However, there is one problem with the traditional machine learning methods. Those methods cannot describe the non-linear relation between independent and dependent variables [2]. However, the neural network can detect all the possible relations between variables [9, 10]. Thus, in this paper, I use the neural network method to do the classification task.

But there is another problem, due to the high dimension of the input feature, some redundant feature may impact the performance of the neural network, it may cause the network to overfit on the old examples and perform badly in the new data [3]. So, feature extraction is needed for this task. And an autoencode is an artificial neural network that uses unsupervised learning to reduce data dimension. Good autoencoder algorithm can introduce sparsity to avoid noise pattern and avoid peaking of weights [12]. Therefore, I use autoencoder method to compress the input features.

Then, inspired by the shared weights auto-associative network in image compression area, I use the shared weights technique in the autoencoder to get a better feature extraction result. Since in a shared weights network, the weights are constrained to be invertible, it will give a better compressed result than the original network [13].

In the experiment part, I design three experiments. Firstly, I investigate the effectiveness of autoencoder. Then, I test the impact of shared weights technique in autoencoder network. Finally, I compare the performance of my model with logistic regression method. The results confirm the importance of autoencoder and the effectiveness of shared weights technique. The results also show the undeniable advantages of my model compared with logistic regression method.

2 Methods

I formulate the task of distinguishing the breast cell from benign or malignant as a classification problem and design a neural network architecture that contains shared weights autoencoder for this problem. The aim of this model is to investigate the performance of neural network, the impact of autoencoder and the effectiveness of shared weights technique. For the dataset, I choose the Breast Cancer Wisconsin(Diagnostic) dataset (section 2.1). And the overview of the network architecture is shown in Fig 1.

As shown in the Fig1, the neural network can be divided in three parts: 1) data preprocessing for dealing with the input data (section 2.2). 2) a shared weights autoencoder for extracting the important features (section 2.3). 3) linear neural network for extracting the type features (section 2.4). The final output of my model is a probability vector for benign and malignant cell (section 2.4). In the training, to fully take use of the finite dataset and get an optimal training performance, I use 10-fold validation method and mini-batch method (section 2.6). And I use two different loss functions for the autoencoder and classification ANN (section 2.5). For evaluation, I choose the accuracy and F1 score (section 2.7).

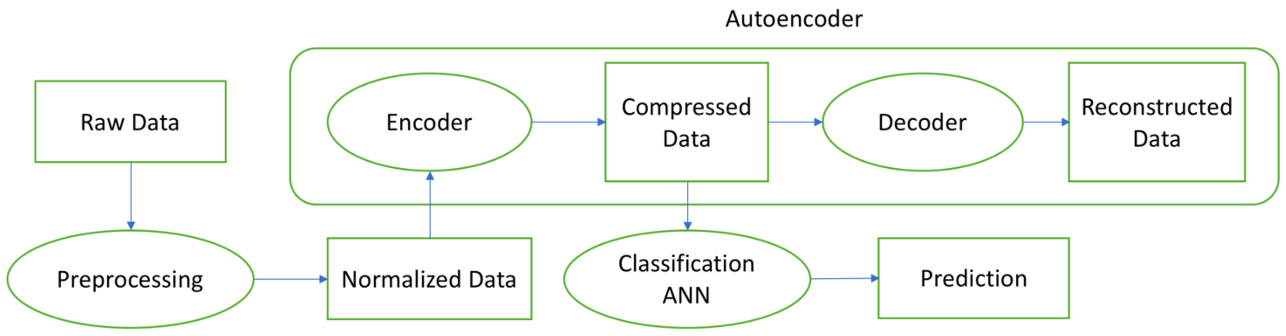

Fig. 1. Overview of the network architecture.

2.1 Dataset Description

In this paper, I choose the Breast Cancer Wisconsin(Diagnostic) dataset. And I choose this dataset for two reasons. Firstly, computer aided methods have become the best choice in cancer prediction area, and this dataset has been widely used for cancer prediction since 1990s [1,6]. Therefore, this dataset is of high quality. Secondly, the features in this dataset are well detected. All the features are converted from digital images that are collected from fine needle aspiration(FNA) [1], and there is no missing value.

This dataset contains 569 cell samples: 212 are benign and 357 are malignant. Each sample has 32 features, first one is the cell id, the second is the cell type (malignant or benign). The other features are real-value features computed from each cell nucleus, such as radius, texture and perimeter.

2.2 Data Preprocessing

In the raw data, most of the features are all number. The only non-number feature is the cell labels. For the cell label feature, the malignant cell is labelled with "M", and the benign cell is labelled with "B". So firstly, I convert the non-number feature to numbers by replacing "M" with 0 and "B" with 1 for all the data. Then, I use the converted data that contains only numbers to do the normalization task.

When we use neural network to do the classification task, the data normalization can greatly influence the training process. Sola and Sevilla [14] shown that in a neural network model, the importance of data normalization lies on accelerating the training process and improving the result quality. Jayalakshmi and Santhakumaran [15] also pointed out that normalization methods can make the feed forward backpropagation more reliable and the diabetes neural network model can give a better result with normalization technique. And in this dataset, some features are very small, some others are very large. So, in this paper, I choose one commonly used normalization method, that is the normal distribution method.

Given certain new mean value and deviation value, the normal distribution method can project the original dataset to a new interval (see Fig.2).

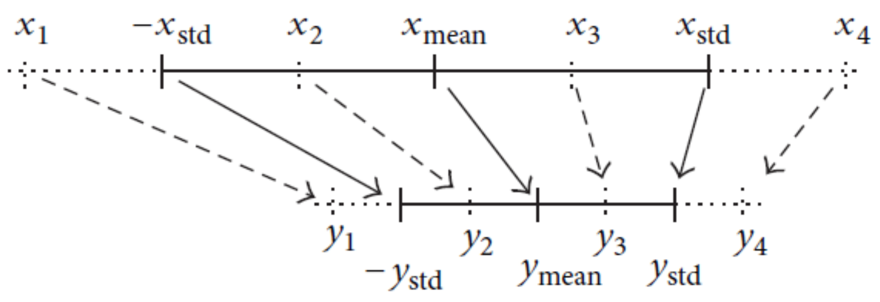

Fig. 2. Normal distribution method of normalization.

And the following formula for the calculation, the original mean value x_{mean} and distribution value x_{std} can be calculated from the raw data. The y_{mean} and y_{std} are the new mean value and distribution value assigned by us.

$$y = \frac{(x - x_{mean}) \cdot y_{std}}{x_{std}} + y_{mean} \quad (1)$$

In my model, the normalized data can be used as the input data directly. Because the normalized data can avoid the peak in weights and reduce the training time. During the training process, the weights can get to the optimal value smoothly with normalized data. There is no need for extra encoding techniques.

2.3 Autoencoder and Shared Weights Technique

In machine learning area, feature extraction is an important approach to reduce the redundant features and select the more meaningful features. A meaningful feature representation can highly improve the accuracy for classification task [16]. In this article, instead of using traditional feature selection methods, I use one popular deep learning method, that is autoencoder.

An autoencoder is a multilayer network that can contains an input layer, some hidden layer and an output layer. The aim of the hidden layer is to reconstruct the input data, so the hidden layer contains all the essential information from input layer. As shown in Fig.3, the encoder is this the compress process from input to hidden layer, the decoder is the output layer from hidden layer to output layer. The encoder and decoder are inverse to each other, thus, the autoencoder architecture is a symmetrical mode. After training, the encoder can be used for extracting features from raw data. In my model, the preprocessed data is used in the autoencoder for data compression.

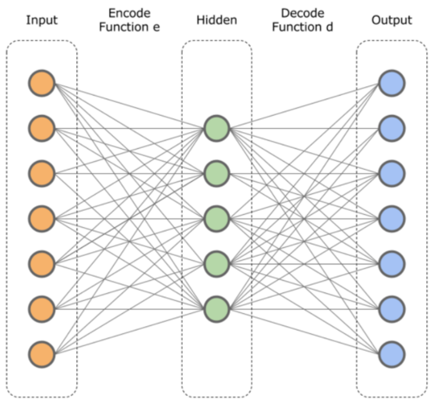

Fig. 3. The structure for autoencoder.

The idea of the shared weights technique comes from the associative neural network in image compression task. The associative neural network is like the standard feed-forward network, expect that the connection weights keep the same between the input layer to hidden layer and hidden layer to output layer. And, the shared weights associative neural network should have a better performance than normal associative network. Since in a normal network, the first layer is a compressed function and the second layer is a decomposed function. But the shared weights network has a stricter constrain. The first and second layers share the same weight, so these two functions are invertible, and the network has less free parameters. Intuitively, we can expect that the shared weights network should have a better performance than the normal one, because the inverse function is “the” function rather than an approximation [13]. So, in my model, I combine the autoencoder and shared weights technique into shared weights autoencoder, and this shared weights autoencoder should have a better performance than the normal autoencoder.

2.4 Artificial Neural Network(ANN)

In the classification network part, I use the linear artificial neural network. Human brain is like a high complex computer that responses to different non-linear variables [17]. Inspired by the human brain structure, people invent the ANN computer model. Like biological neural network, ANNs contain scores of interconnected nodes that process inputs from other nodes [18]. ANNs can be regarded as a black box that contains lots of weights for calculating the model output [19]. And the overall performance of a ANN is determined by the connections between different nodes. Although ANNs have various approaches for connecting the nodes and processing inputs, I just focus on the normal “feedforward” network in my model.

The Fig.4 shows the structure for the classification artificial neural network. The basic ANN contains three layers (input layer, hidden layer and output layer), each layer contains some nodes. In the input layer, the nodes are called input nodes, and they represent the input variable (in this paper, that is a compressed feature from shared weights encoder). In the output layer, each node means a predicted result (probability for belonging cell classes). In the hidden layer, the hidden nodes do not have actual meaning, they are used to model the interconnection between input nodes and the relations between input features and output result. And in this Fig.4, there are n input nodes, k hidden nodes and 2 output nodes. And there are some arcs, they represent the connection weights between nodes. Mathematically, we can describe a neural value by formula (2) and (3).

$$u_k = \sum_{j=1}^{n} w_{kj} x_j \quad (2)$$

$$h_k = \theta(u_k + b_k) \tag{3}$$

In the formula (2) and formula (3), x_j is the input (In autoencoder part, the input is preprocessed data. In classification neural network, the input is compressed data). w_{kj} is the weight between input x_j and the node k. u_k is the linear combination output. b_k is the bias, which is used to adjust the input value for activation function. θ is the activitaion function that is used to adjust the weights in multilayer neural network. Usually, people use sigmoid function in the hidden layer and linear function in the output layer. In the autoencoder part, I use sigmoid function for each layer. In the classification network, I use sigmoid function for hidden layers and softmax function for output layer. h_k is the real output for node k.

With labeled data, the ANNs can learn the best weights that can describe the relation between inputs and outputs by changing the weights values. This process is called training. There are various training algorithms, and the most common one is backpropagation. The main idea for backpropagation is changing the weights in each training to minimize the difference between real label and predicted label [20]. In my model, I choose the backpropagation for the classification task.

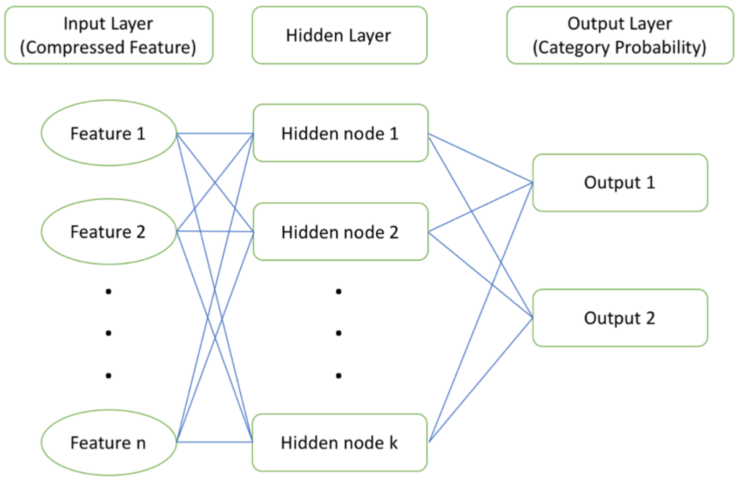

Fig. 4. The structure of classification neural network

In my model, the output layer has two nodes. And I use the softmax function for the output layer activation function. The softmax function can constrain the arbitrary value in a vector to range (0,1), and the sum of the values are 1 [25]. The softmax function is shown in formula (4). In here, z is the hidden value, K is the class number, $\sigma(z)_j$ is the output value.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{4}$$

As for the reasons of choosing softmax as decoder, it is because of the property of softmax function. Firstly, softmax function can map the output value to range (0,1). Secondly, the value can be added up to one. So, after using softmax, the two outputs can represent the probability of being benign and malignant.

2.5 Loss function

In backpropagation, the loss function calculates the difference between the predicted output and the expected output. In my model, I use two different loss functions for the autoencoder and classification network. For the autoencoder, I use the commonly used loss function, which is mean square error (MSE). It can be calculated by the formula (5). The x_i is the input data, the $G(x_i)$ is the predicted label, and y_i is the real label. It can calculate the mean distance between predicted output and real label.

$$\mathrm{C} = \frac{1}{n}\sum_{i=1}^{n}(G(x_i) - y_i)^2 \tag{5}$$

For the classification network, I use the cross-entropy method. Because in classification task, the cross-entropy has a better performance than square error method [22]. It can be calculated by formula (6). To minimize the cost C, when label y_i is 0, $y_i \ln G(x_i)$ is 0, we will try to make $G(x_i) \approx 0$, so $(1 - y_i)\ln(1 - G(x_i)) \approx 0$. When label y_i is 1, vise verse.

$$\mathrm{C} = -\frac{1}{n}\sum_{i=1}^{n}(y_i \ln G(x_i) + (1 - y_i)\ln(1 - G(x_i))) \tag{6}$$

2.6 10-fold Cross-Validation and Mini-Batch Method

In my model, I take the 10-fold cross-validation method to split the train and test data. The process is like following. Firstly, I divide the dataset in to 10 equal parts. Then I take one part as the test set, the other parts as training set. After the training and test, I continue this process until all parts have been tested. By 10-fold cross-validation, I can use all data in both training and test process. Thus, I can fully use the dataset and get an accurate and unbiased result. What is more, I use the mini-batch gradient decent method. The mini-batch method is an algorithm that can split the training dataset into small batches [26]. And each batch is used to calculate the model error and update the model weights. Using this method, my model can get a more robust convergence and avoid the local minimal.

2.7 Evaluation Method

In classification task, it is important to evaluate the model performance. And accuracy and F1 score is the most common evaluation value. In this paper, I take both accuracy and F1 score as the evaluation value.

Table 1. Definition for TP, TN, FP and FN

	Class=Yes (Predicted)	Class=No (Predicted)
Class=Yes (Actual)	True Positive (TP)	False Negative (FN)
Class=No (Actual)	False Positive (FP)	Ture Negative (TN)

To calculate the F1 score, we have to know the four parameters in table 1:

True Positive(TP): the correctly predicted positive number, that means both the actual class and predicted class are yes.

True Negative(TN): the correctly predicted negative number, that means both the actual class and predicted class are no.

False Positive(FP): the falsely predicted positive number, that means the actual class is no, but predicted value is yes.

False Negative(FN): the falsely predicted negative number, that means the actual class is yes, but the predicted value is no.

Then we can calculate the precision value and recall value. Precision is the ratio of correctly predicted positive data to the total predicted positive data. It shows the exactness of a classifier. The formula (8) shows the calculation for precision.

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (7)$$

Recall is the ratio of correctly predicted positive data to the actual positive data. It shows the completeness of a classifier. The formula (8) shows the calculation for recall.

$$\text{Recall} = \frac{TP}{TP+FN} \qquad (8)$$

The F1 score can be calculated as the formula (9), it can keep a balance of precision and recall. Thus, it can show both the exactness and completeness of a classifier.

$$F_1 = 2 \cdot \frac{precision \cdot \mathrm{rec}all}{precision+recall} \qquad (9)$$

Especially for k-fold validation, there are various ways to calculate F1 score. The best way to is computing TP, FP and FN for each fold, and calculate the F1 score based on the “micro” metrics. That means we can sum the TP, FP and FN from each fold, and use the sum value to calculate F1. Because it can give us the most unbiased result [21].

Accuracy is the ratio of correctly predicted observation to the total observation. It is used to describe how accurate the model is. It can be calculated by the formula (10).

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \qquad (10)$$

In the evaluation part of my model, I consider both accuracy and F1 score. Therefore, I can not only evaluate the exactness and completeness of the classifier, I can also evaluate how accurate the model is.

3 Result and Discussion

In this part, I conduct three experiments to answer the following three questions:

Question1: What is the impact of autoencoder? Will the performance of artificial neural network improve after adding autoencoder?

Question 2: How effective is the shared weights for autoencoder? Will shared weights technique improve the feature extracting ability?

Question 3: How well can my model do the classification task compared with logistic regression?

3.1 Impact of the autoencoder

Motivation: Recently, autoencoder has been used to do the feature extraction task, and shows better performance than traditional feature extraction method, such as PCA. The autoencoder is a good choice to deal with the redundant feature problem. In the breast cell classification problem, some features are redundant. Since the dataset was collected from video images, each cell may have different shape in different images. Some fields in the dataset describe the same feature. For example, the feature field 3 describe the Mean radius, field 13 describe the radius standard error, and field 23 describes the largest radius. The field 3, 13 and 23 all describe the radius feature. To reduce the redundant feature problem, I use autoencoder to do the feature extraction task. And the first problem is to confirm the effectiveness of autoencoder by comparing one ANN that contains no autoencoder and one ANN that uses autoencoder.

Approach: To test the effectiveness of autoencoder, I design two models. The first model is a ANN that uses autoencoder to compress features. It contains two parts, one is normal autoencoder, the other one is a normal ANN. The autoencoder contains one hidden layer with 10 nodes, the classification ANN part contains one hidden layer with 5 nodes. The second model is a normal ANN. To make a fair comparison, the autoencoder parameters are also trained in the classification ANN training. What is more, the second model has a similar structure with the first one. It contains two hidden layers that have 10 nodes and 5 nodes. Then to test the impact of the autoencoder. Firstly, I train the autoencoder of the first model for 300 epochs. Then the classification network of the first model and the second model are trained at same epochs. I compare the accuracy and F1 value of the two models at different epochs for classification ANN.

Table 2. The accuracy and F1 score for three different models at different epochs for classification network

Epoch	Basic ANN		ANN with pre-trained normal autoencoder		ANN with pre-trained shared weights autoencoder	
	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
1	78.38	59.67	85.76	81.80	91.56	87.94
2	89.81	84.32	95.61	93.98	95.78	94.09
3	90.51	85.56	96.66	95.47	96.31	94.92
4	90.86	86.10	97.72	96.88	97.19	96.12
5	96.31	94.82	97.54	96.64	97.89	97.12
6	97.89	98.09	97.72	96.88	98.07	97.38
7	98.77	98.32	98.24	97.61	98.24	97.61
8	97.89	98.09	98.07	97.36	98.24	97.61
9	98.59	98.08	98.42	97.84	98.24	97.61
10	98.77	98.32	98.24	97.61	98.07	97.35

Result and discussion: The Table 2 shows the accuracy and F1 value of the two models at different epochs for classification network. From this table, we can see that after long time training, the first model and second model both has a high accuracy and F1 value. At the epoch 10, the accuracy and F1 value for basic ANN is 98.77 and 98.32, the accuracy and F1 value for the ANN with autoencoder is 98.24 and 97.61. But in a shot training time, the ANN model that contains autoencoder shows a much better result than the basic ANN model. At epoch 1, the accuracy and F1 value for the basic ANN model is just 78.38 and 59.67, the accuracy and F1 value for the other model is 85.76 and 81.80. The accuracy is 9% higher, and the F1 value is 37% higher. At epoch2, the accuracy of second model is 6% higher than the basic ANN and the F1 value is 11% higher.
The reason may be that with autoencoder, redundant feature will not mislead the model to a wrong direction and noise data can be ignored, thus the model can learn a good result in a shorter time. The autoencoder network can approximate a pair of invertible functions. During training, the encoder is a compress function, it can compress the features in hidden layer. And the decoder is a decompress function, it aims to reconstruct the original data [23]. After the training, the encoder has the ability of extracting important information from raw data. It can also reduce the training time for furthermore classification task.

3.2 Effectiveness of the shared weights technique

Motivation: The shared weights technique has been used in image compression and decompression task for a long time. With the shared weights technique, although the image quality cannot improve, the robustness is improved significantly [13]. That inspired me to use the share weights technique in the autoencoder. Hopefully, the model that uses shared weights autoencoder can show the similar robustness. And the second problem is to test the impacts of the shared weights technique.

Approach: To test the impact of share weights technique, I compare two models. The first model uses normal autoencoder. The seconder model uses shared weights autoencoder. The two model both contain two parts, one is autoencoder, the other one is classification network. And these two networks are trained separately. Then I hold two experiments. In the first experiment, I keep the autoencoder training time unchanged, and compare the performance of two models when the classification network training time changes. And in the classification network training, the autoencoder parameters are not frozen. In the second experiment, I keep the classification network training time unchanged at epoch 5 and compare the performance of two models when the autoencoder training time changes. In this experiment, the autoencoder parameters are frozen. These two experiments is to test whether shared weights technique improve the feature extraction ability.

Table 3. The accuracy and F1 score for the two models at different autoencoder epoch

Epoch	ANN with normal autoencoder		ANN with shared weights autoencoder	
	Accuracy	F1-score	Accuracy	F1-score
1	68.19	29.57	94.90	92.91
2	75.57	51.57	93.85	91.53
3	77.86	60.38	96.13	94.67
4	79.61	62.58	96.31	94.87
5	77.15	57.72	96.60	95.38
10	83.13	73.33	96.66	95.44
50	92.97	89.74	95.61	93.92
100	92.62	89.01	95.78	94.18
500	95.08	93.14	94.03	91.50

Result and discussion: From the Table 2, we can see that when the autoencoder training time is unchanged, the performance of shared weights model better than the normal one in short training time. At epoch 1, the accuracy and F1 score for shared weights model is 91.56 and 87.94, the accuracy and F1 score for normal model is just 85.76 and 81.8. That indicates that the shared weights technique improves the classification task performance in limited training time. As for the reason, I think it is because that in a long training time, the shared weights autoencoder and the normal autoencoder can both extract the important features. But in a short time, the shared weights technique can improve the feature extraction ability of autoencoder.

In the Table 3, the classification network training time is unchanged and the autoencoder parameters are frozen. When the autoencoder training time is long, these two model both have good performance. But when the autoencoder training time is short, the shared weights autoencoder model can give a better performance. At epoch 1, the accuracy and F1 score for the normal autoencoder model is just 68.9 and 29.57, the accuracy and F1 value for the share weights model is 93.85 and 91.53. The performance of the shared weights model overwhelmly overcome the normal one. This indicates that the shared weights technique can improve the efficiency and robustness of the feature extraction. I guess that it is because that the shared weights autoencoder is invertible in compress and decompress process. The shared weights technique forces the compress function and decompress function to be the unique pair. Hence, the shared weights autoencoder should perform better than the normal autoencoder, because the normal autoencoder is just to approximate the unique invertible function [13].

3.3 Comparison with logistic regression method

Motivation: logistic regression method shows a good performance in the breast cell classification task and has been widely used for the breast cell classification baseline. Thus, in this part, I want to investigate the performance of my model compared to this baseline.
Approach: To compare the performance, I take the optimum accuracy and F1 score for my model and compare with the logistic regression method results from W. H. Wolberg et al [1].

Table 4. The confusion matrix for logistic regression method

	Malignant, Confirmed	Benign, Confirmed	Total
Malignant, Predicted	199.6±0.65	9.0±0.72	208

Benign, Predicted	12.4±0.65	348.0±0.72	361
Total	212	357	569

Table 5. The confusion matrix for ANN that contains shared weights autoencoder

	Malignant, Confirmed	Benign, Confirmed	Total
Malignant, Predicted	203	2	205
Benign, Predicted	9	355	364
Total	212	357	569

Result and discussion: From Table 4 and 5, we can see that the accuracy and F1 score of my model is much higher than the logistic regression, the accuracy and F1 score of my model are 98.07 and 97.36. And the accuracy and F1 score for logistic regression is 96.2 and 94.91. That shows my model has a better performance than the logistic regression model. The high performance of my model may be related to the following two reasons. First of all, the logistic regression cannot deal with non-linear relations, but neural network has the ability of making association between nonlinear parameter by taking them as proportional weights. For instance, ANN can deal with the mass size parameters that have no detectable correlation, the dependent relation with other parameters are non-zero weights in ANN [24]. Secondly, autoencoder can avoid noise data and extract important feature. Using the compressed feature, the classification network can be highly accelerated. What is more, the shared weights technique can improve the feature extraction ability for autoencoder. So, my model should perform better than the logistic regression model.

4 Conclusion and Future Work

This paper represents a novel neural network that uses shared weights autoencoder to effectively extract important feature. With autoencoder, the needed training time for the neural network is reduced significantly. Because autoencoder has the ability of extracting important feature and ignoring noise data. Furthermore, with shared weights technique, the autoencoder appears more robust, the autoencoder can learn the compression ability in a much shorter time. And this neural network shows better performance than the logistical regression method.
In the future, I will try some innovative technique with my model. In the autoencoder part, I will test the performance of bidirectional autoencoder. Because in image compression task, with different bidirectional technique, the network shows better performance [13]. And I will try to combine the neural network and traditional machine learning method. For example, I can use autoencoder to compress raw feature, then use logistic regression method to do the classification task.

References

[1] W. H. Wolberg et al, "Computer-derived nuclear features distinguish malignant from benign breast cytology," Human Pathology, vol. 26, (7), pp. 792-796, 1995.

[2] J. V. Tu, "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes," Journal of Clinical Epidemiology, vol. 49, (11), pp. 1225-1231, 1996.

[3] Z. M. Hira and D. F. Gillies, "A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data," Advances in Bioinformatics, vol. 2015, pp. 1-13, 2015.

[4] Quinlan, J.R. (1993). C4.5: Programs for Machine Learning. San Mateo, CA: Morgan Kaufmann.

[5] Elkan, C. Boosting and Naïve Bayesian Learning. Technical Report No. CS97-557, Department of Computer Science and Engineering, University of California, San Diego, Spetember 1997.

[6] W. H. Wolberg, W. N. Street and O. L. Mangasarian, "Machine learning techniques to diagnose breast cancer from image-processed nuclear features of fine needle aspirates,"Cancer Letters, vol. 77, (2), pp. 163-171, 1994.

[7] G. R. M. A. Sizilio et al, "Fuzzy method for pre-diagnosis of breast cancer from the Fine Needle Aspirate analysis," BioMedical Engineering Online, vol. 11, (1), pp. 83-83, 2012.

[8] Y. M. George et al, "Breast Fine Needle Tumor Classification using Neural Networks,"International Journal of Computer Science Issues (IJCSI), vol. 9, (5), pp. 247, 2012.

[9] G. Hinton, "How Neural Networks Learn from Experience", Scientific American, vol. 267, no. 3, pp. 144-151, 1992.

[10] H. White, "Learning in Artificial Neural Networks: A Statistical Perspective", Neural Computation, vol. 1, no. 4, pp. 425-464, 1989.

[11] J. V. Tu, "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes," Journal of Clinical Epidemiology, vol. 49, (11), pp. 1225-1231, 1996.

[12] A. Sankaran et al, "Group sparse autoencoder," Image and Vision Computing, 2017.

Cancel

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

[14] J. Sola and J. Sevilla, "Importance of input data normalization for the application of neural networks to complex industrial problems," IEEE Transactions on Nuclear Science, vol. 44, (3), pp. 1464-1468, 1997.

[15]T. Jayalakshmi and A. Santhakumaran, "Statistical Normalization and Back Propagationfor Classification", International Journal of Computer Theory and Engineering, pp. 89-93, 2011.

[16] A. Sankaran et al, "Group sparse autoencoder," Image and Vision Computing, 2017.

[17] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall, Ontario, 1999.

[18] R. Ata, "Artificial neural networks applications in wind energy systems: a review,"Renewable and Sustainable Energy Reviews, vol. 49, pp. 534-562, 2015.

[19] M. Ozgoren, M. Bilgili and B. Sahin, "Estimation of global solar radiation using ANN over Turkey," Expert Systems with Applications, vol. 39, (5), pp. 5043-5051, 2012.

[20] S. M. Kim et al, "A Comparison of Logistic Regression Analysis and an Artificial Neural Network Using the BI-RADS Lexicon for Ultrasonography in Conjunction with Introbserver Variability," Journal of Digital Imaging, vol. 25, (5), pp. 599-606, 2012.

[21] G. Forman and M. Scholz, "Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement," ACM SIGKDD Explorations Newsletter, vol. 12, (1), pp. 49-57, 2010.

[22] D. M. Kline and V. L. Berardi, "Revisiting squared-error and cross-entropy functions for training neural network classifiers," Neural Computing and Applications, vol. 14, (4), pp. 310-318, 2005.

[23] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (5) (1989) 359–366.

[24] Yarmohammadi, M., Abdolmaleki, P., & Gity, M. (2004). Comparison of logistic regression and neural network models in predicting the outcome of biopsy in breast cancer from MRI findings. Iranian Journal of Radiation Research, 1(4), 217-228.qiqi

[25] D. Yu, "Softmax function based intuitionistic fuzzy multi-criteria decision making and applications," Operational Research, vol. 16, (2), pp. 327-348, 2016.

[26] M. Li, T. Zhang, Y. Chen, and A. J. Smola. Efficient minibatch training for stochastic optimization. In ACM KDD, 2014.