Using Genetic Algorithm-based Feature Selection and Shared Weights Autoencoder in Neural Network for Breast Cancer Analysis

Author Name (Paper 195 assignment 2)

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. Traditional machine learning methods are often used to analyze the cell information and classify the cell type. In this paper, two new neural network models will be presented for this classification task. One is the combination of artificial neural network(ANN) and shared weights autoencoder, and the other one contains both ANN and genetic algorithm(GA)-based feature selection. In these models, the ANN is used for classification. Shared weights autoencoder and GA-based feature selection are used to solve the redundant features problem. Furthermore, I conduct several experiments with the Breast Cancer Wisconsin(Diagnostic) dataset that has well collected data. My experiments investigate the impact of autoencoder, the effectiveness of shared weights technique and the impact of GA-based feature selection. I also analyze the performance of my two models and logistic regression model. My results confirm the feature extraction ability of autoencoder and the effectiveness of shared weights technique. What is more, the results show the superior performance of my two models compared with logistic regression model. And GA-based feature selection performs better than shared weights autoencoder but cost more computation resources.

Keywords: Neural network, Shared weights, Autoencoder, Genetic algorithm, Breast cancer, Feature extraction, Feature selection

1 Introduction

As an essential part in both cancer treatment and cancer prevention, cancer prediction is an approach using different bio-information to predict whether a cell is malignant. With the aid of computers, microscope image can be converted to accurate feature information, which can highly improve prediction accuracy and objective feature assessment [6].

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, such as logistic regression [1], C4.5 decision trees [4] and Naïve Bayesian learning [5].

In the previous work, different methods have been investigated on this cancer prediction task. W. H. Wolberg et al use logistic regression and inductive machine learning method, which provide the base for breast cell analysis [1]. G. R. M. A. Sizilio et al use fuzzy system, which can reduce the variation hit of malignant cases [7]. Y. M. George et al test the performance of SVM, which 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 gains the accuracy with the value of 96.2% [1]. However, problems exist in traditional methods drive the design of two models used in the experiments. One problem is that traditional methods cannot describe the non-linear relation between independent and dependent variables [2]. Because the neural network can detect all the possible relations between variables [9, 10]. Neural network method can be used in classification task to solve the problem stated above. The other problem is that input features have large dimensions. Some redundant features may impact the performance of the neural network and lead the network to overfit on the old examples but perform badly in the new data [3]. Autoencoder can be used for feature extraction in this task. An autoencoder is an artificial neural network that uses unsupervised learning to reduce data dimension, which can be used for feature extraction. Good autoencoder algorithm can also avoid noise pattern and peaking of weights [12]. Besides, 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 the weights in a shared weights network must be invertible, it will give a better compressed result than the original network [13]. Therefore, ANN and shared weight autoencoder were used in my first model to do classification and feature extraction respectively.

However, based on further investigation, feature extraction method often deletes some discriminative features and reduce the verification performance [24]. On the other hand, some extracted features may have no effects in the performance improvement [25]. Thus, feature selection method needs to extract useful and discriminative features. Genetic algorithm is a powerful tool for feature selection, especially when the input feature dimension is large [26]. By reducing the feature dimensions, genetic algorithm can not only reduce the computation complexity but also improve the performance of classifier in classification task [27]. Therefore, the second model contains both GA-based feature selection and ANN with possibility of potential improvement.

The structure of this paper is organized as following. Section 2 describes the methods used in my two models. Section 3 summarizes the experimental results and discussions. Conclusions and the direction for future work are given in Section 4.

2 Methodologies

I formulate the task of distinguishing the breast cell from benign or malignant as a classification problem and design two neural network architectures for this problem. This first model contains ANN and shared weights autoencoder. The aim of this model is to investigate the performance of neural network, the impact of autoencoder and the effectiveness of shared weights technique. The second model contains ANN and GA-based feature selection. This model is designed to investigate the impact of GA-based feature selection. The Breast Cancer Wisconsin (Diagnostic) dataset is chosen as the dataset for training (section 2.1). The overview of the two network models is shown in Fig.1 and Fig.2 respectively.

As shown in the Fig.1, the first model 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) artificial neural network for classification (section 2.4). The Fig.2 shows the structure of the second model. The input preprocessing and classification parts are the same as the first model. To select important features, this model uses a GA-based feature selection method instead of autoencoder (section 2.5).

The final output of my models is a probability vector for benign and malignant cell (section 2.4). In the classification ANN training, 10-fold validation and mini-batch were used to make best use of the finite dataset and get an optimal training performance (section 2.6). Two different loss functions were taken for the autoencoder and classification ANN respectively (section 2.5). Both accuracy and F1 score were chosen as the measures for the evaluation (section 2.7).

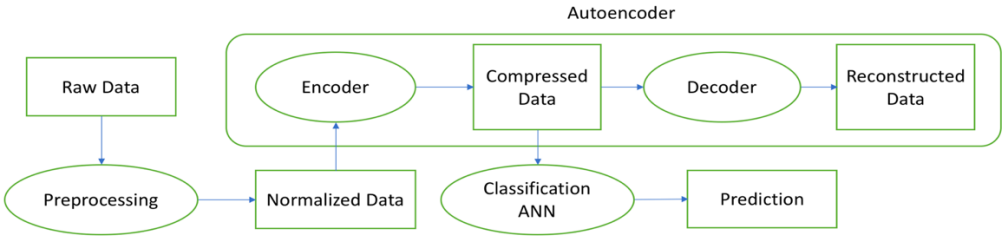

Fig. 1. Overview of the model that contains shared weights autoencoder and ANN.

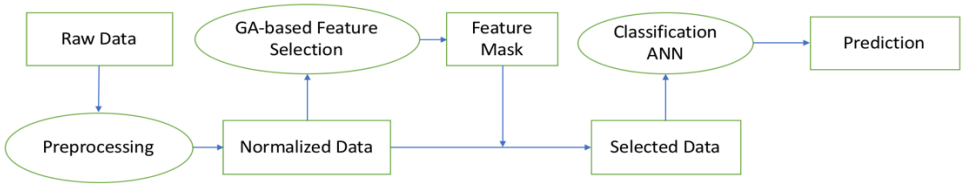

Fig. 2. Overview of the model that contains GA-based feature selection and ANN.

2.1 Dataset Description

In this paper, I choose the Breast Cancer Wisconsin (Diagnostic) dataset with 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 reliable for cancer prediction task. 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, the first feature is the cell id, the second feature 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 numerical, while the only non-number feature is the cell label. For the cell label feature, the malignant cell is labelled with “M”, and the benign cell is labelled with “B”. For the ease of processing, cell labels were converted to numbers by replacing “M” with 0 and “B” with 1 and all data were normalized between 0 to 1.

When neural network is used for classification, the data normalization can greatly influence the training process. Sola and Sevilla [14] have shown that the importance of data normalization lies on accelerating the training process and improving the result quality in a neural network model. Jayalakshmi and Santhakumaran [15] also pointed out that normalization can make the feed forward backpropagation more reliable and the diabetes neural network model can give a better result with normalization technique. Within the dataset, the values of some features are small, while other values are large. Therefore, normalized input data was used for 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.3).

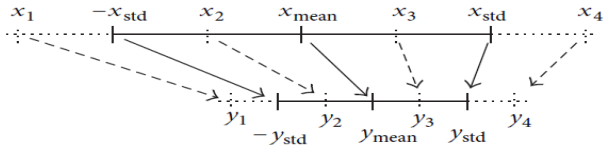

Fig. 3. Normal distribution method of normalization.

Based on the following formula, 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.

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

The normalized data was 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 meaningful features. A meaningful feature representation can highly improve the accuracy for classification task [16]. Instead of using traditional feature selection methods, I use one popular deep learning method, which is autoencoder.

An autoencoder is a multilayer network that can contains an input layer, some hidden layers 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.4, 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 model. After training, the encoder can be used for extracting features from raw data. In my first model, the preprocessed data is used in the autoencoder for data compression.

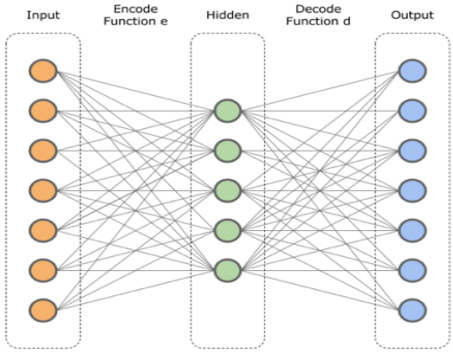

Fig. 4. The structure for autoencoder.

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. 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. The shared weights network has a stricter constrain, which is 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]. Thus, I combine the autoencoder and shared weights technique into shared weights autoencoder, which should have a better performance than the normal autoencoder. Besides, sigmoid activation is used in each layer.

2.4 Classification Artificial Neural Network

In the classification network part, a three-layer artificial neural network was used. It contains 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 (compressed features or selected features). 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.

With labeled data, the ANN can learn the best weights that can describe the relation between inputs and outputs. This process is called training. Backpropagation is used for the training. The main idea for backpropagation is changing the weights in each training to minimize the difference between real label and predicted label [17].

In the classification network, sigmoid function is used in input and hidden layers, and softmax function is used for output layer. The softmax function can constrain the arbitrary value in a vector to range (0,1), and the sum of the values

are 1 [22]. The softmax function is shown in formula (2). In here, z is the hidden value, K is the class number, $\sigma(z)_j$ is the output value.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad (2)$$

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 Genetic Algorithm-based Feature selection

Genetic algorithm, developed by Holland, is a computational optimization procedure inspired from the concept of biological evolution [28]. Genetic algorithm aims to find the best solution in the binary search spaces and manipulates scores of potential solutions [29]. The genetic algorithm simulates the main steps in biological evolution such as reproduction, crossover and mutation, etc. Each possible solution, called a chromosome, is represented by a finite sequence that only contains 0 and 1. The simulated environment, called fitness function, is used to evaluate the quality of a solution. The possibility of survival is related to the fitness value. To select the best features for classifier, I use the genetic algorithm to select informative features and reduce redundant features.

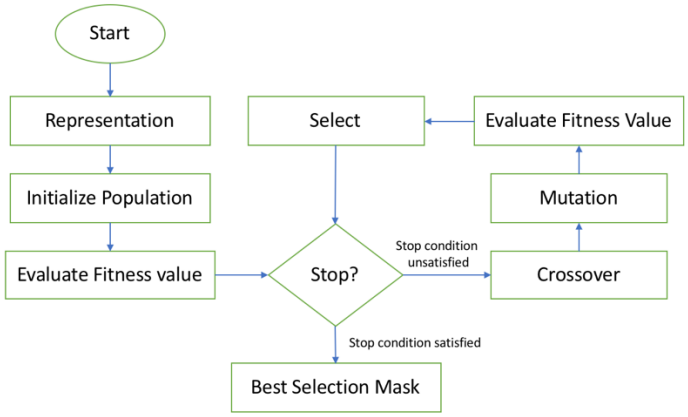

Fig. 5. The flow of the GA-based feature selection

The process of GA-based feature selection is shown in Fig.5, and the main concepts are as following.

Representation: In this feature selection task, each chromosome is defined as a mask for features. For instance, "0" means the related feature is not selected, "1" means the related feature is selected. The process of selecting features with feature mask is shown in Fig. 6.

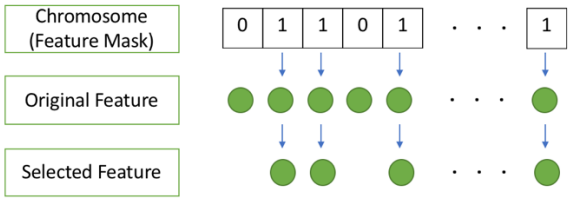

Fig. 6. The process of selecting features with feature mask

Fitness function: To evaluate the fitness value of a chromosome, neural network approach is used. The neural network model is the same as the classifier. This process can be divided in three steps: (1) Using the chromosome to get the masked dataset. (2) Using 10-fold validation to train the network model with masked dataset. (3) Calculating the F1 score as the fitness value.

Selection: Proportional selection is used. During the selection, an empty population is initialized and following process is repeated. An individual will be picked from existing population to new population, and the possibility is proportional to the fitness value. When the size of new population is equal to the previous population, this iteration will stop, and the selection is completed.

Crossover: Uniform crossover is used. For each individual, a random value will be generated. If the random value is larger than the crossover rate, this individual will keep the same. Otherwise, this individual will continue the following crossover process. Another individual will be randomly picked from the population. After that, the genes of the new individual will randomly overwrite the corresponding genes of the mutated individual.

Mutation: Each chromosome will be picked from the population sequentially. For each gene in one chromosome, a random value will be generated. If the value is larger than the mutation rate, this gene will keep the same. Otherwise, this gene will change from 0 to 1 or from 1 to 0.

Evolution: The phases from selection to mutation are iterated. For each evolution, the best feature set is recorded, and it was compared with the best feature set in all generations. If the current best feature set is better, it is recorded as the

best feature in all generations. If there is no improvement for 5 times, this genetic algorithm is completed. The best feature set in all stages is the selected feature.

2.5 Loss function

In backpropagation, the loss function calculates the difference between the predicted output and the expected output. Two different loss functions were used for the autoencoder and classification network. For the autoencoder, mean square error (MSE) is used. It can be calculated by the formula (3). 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.

$$C=\frac{1}{n}\sum_{i=1}^{n}(G(x_i)-y_i)^2 \quad (3)$$

For the classification network, cross-entropy method is used. Because in classification task, the cross-entropy has a better performance than square error method [19]. It can be calculated by formula (4). 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.

$$C=-\frac{1}{n}\sum_{i=1}^{n}(y_i \ln G(x_i)+(1-y_i)\ln(1-G(x_i))) \quad (4)$$

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

In the classification ANN and evaluation part of GA-based feature selection, 10-fold cross-validation method is used to split the train and test data. The process is like following. Firstly, the dataset is divided in to 10 equal parts. Secondly one part is chosen for testing, and the other parts are kept for training. After the training and testing, this process in continued until all parts are tested. By 10-fold cross-validation, I can use all data in both training and test process and get an accurate and unbiased result.

What is more, mini-batch gradient decent method is used in classification ANN. The mini-batch method is an algorithm that can split the training dataset into small batches [23]. Each batch is used to calculate the error and update the weights. Using this method, the classifier 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. Accuracy and F1 score are used for evaluation.

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. True Negative(TN): the correctly predicted negative number. False Positive(FP): the falsely predicted positive number. False Negative(FN): the falsely predicted negative number.

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. Recall is the ratio of correctly predicted positive data to the actual positive data. It shows the completeness of a classifier. The F1 score can keep a balance of precision and recall. Thus, it can show both the exactness and completeness of a classifier. The formula (5), (6) and (7) shows the calculation for precision, recall and F1 score.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (5)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (6)$$

$$F_1=2 \cdot \frac{precision \cdot recall}{precision+recall} \quad (7)$$

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 [18].

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 (8).

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \quad (8)$$

Therefore, with accuracy and F1 score, I can not only evaluate the exactness and completeness of the classifier but also evaluate how accurate the classifier is.

3 Result and Discussion

Several experiments were conducted to answer the following questions:
1. What is the impact of autoencoder? Will the performance of artificial neural network improve after adding autoencoder?
2. How effective is the shared weights for autoencoder? Will shared weights technique improve the feature extracting ability?
3. What is the impact of GA-based feature selection? Will this method improve the performance of ANN classifier?
4. Does GA-based feature selection perform better than shared weights autoencoder?
5. How is the performance of my two models in the classification task compared with logistic regression?

3.1 Impact of the autoencoder

Motivation: 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 features, autoencoder is used for feature extraction. This experiment is to confirm the effectiveness of autoencoder by comparing normal ANN and the model that uses autoencoder and ANN.
Approach: To test the effectiveness of autoencoder, two models are compared. The first model is a model 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, the autoencoder of the first model is trained for 300 epochs. After that, 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	73.64	45.65	85.76	81.80	91.56	87.94
2	86.12	77.36	95.61	93.98	95.78	94.09
3	86.82	78.76	96.66	95.47	96.31	94.92
4	90.33	85.26	97.72	96.88	97.19	96.12
5	97.01	95.84	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	98.59	98.08	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 model with autoencoder performs slightly worse than normal ANN. 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 short 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 73.64% and 45.65%, the accuracy and F1 value for the other model is 85.76% and 81.80%. The accuracy is 16% higher, and the F1 value is 88% higher. At epoch 2, the accuracy of second model is 11% higher than the basic ANN and the F1 value is 21% higher.

The reason may be that with autoencoder, redundant features will not mislead the model to a wrong direction and noisy 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 [20]. After the training, the encoder has the ability of extracting important information from raw data. The extracted features can help classifier perform well in limited training time. However, the autoencoder may also ignore some discriminative features. The ignorance of discriminative features may lead to the worse performance in a long training time.

3.2 Effectiveness of the shared weights technique

Motivation: In image compression and decompression task, although the image quality cannot improve with the shared weights technique, the robustness improves significantly [13]. That inspires me to use the share weights technique in the

autoencoder. Hopefully, the model that uses shared weights autoencoder can show the similar robustness. This experiment is to test the impacts of the shared weights technique.

Approach: To test the impact of share weights technique, two models are compared. The first model uses normal autoencoder. The seconder model uses shared weights autoencoder. These two models both contain two parts. One is autoencoder, the other one is classification network. Two experiments are conducted for these two models. In the first experiment, autoencoder training time are unchanged at 300 epochs, and the performances of two models are compared when the classification network training time changes. Besides, the autoencoder parameters are not frozen in the classification network training. 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 in the classification network training.

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. In a short time, the shared weights technique can improve the feature extraction ability of autoencoder. Thus, it can help classifier have better performance in limited training time.

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, and the normal autoencoder is just to approximate the unique invertible function [13]. Hence, the shared weights autoencoder should perform better than the normal autoencoder.

3.3 Impact of the GA-based feature selection

Motivation: When the original features are compressed by autoencoder, some noisy features are ignored. However, some discriminative features may also be deleted in this process and the performance of classifier may decrease. Thus, feature selection is needed to solve this problem. Besides, genetic algorithm has been used for feature selection for a long time. Many literatures have shown the undeniable advantages of genetic algorithm in feature selection task [30], [31], [32]. Therefore, GA-based feature selection is used in my second model. Hopefully, GA-based feature selection could improve the performance of ANN classifier and reduce the computation cost. This experiment is to investigate the impact of GA-based feature selection.

Approach: To test the performance of GA-based feature selection, two models are compared. The first model is a normal ANN classifier. The second model uses a GA-based feature selection to get the important features then uses a ANN to classify the cell type. For GA-based feature selection, the crossover rate is 0.8 and the mutation rate is 0.002, which ensure the diversity. To make a fair comparison, these two models has same number of hidden units in ANN classifier, 10 units in first hidden layer, 5 units in second hidden layer. Two experiments are conducted to test the performance of these two models. In the first experiment, the dataset is the original dataset. Firstly, the GA-based feature selection is used to select the feature mask. Secondly, the classification of the second model and the first model are trained at same epoch. The accuracy, F1 socre and time cost are compared at different epoch. Since the feature size is too small, I add extra noisy features to the original dataset to test the selection ability of GA-based feature selection. The feature size is extended from

30 to 1000. Then I use the noised dataset to hold another experiment, this experiment process is same as the first one. Besides, the used computer is Apple MacBook Pro 13.3 with 3.1 GHz Intel Core i5 processor, 8 GB memory and Intel Iris Plus Graphics 650 1536 MB graphics.

Table 4. The accuracy, F1 score and time cost for the two models at different epochs with different dataset

Epoch	ANN (original dataset)			ANN with GA-based feature selection (original dataset)			ANN (noised dataset)			ANN with GA-based feature selection (noised dataset)		
	Accuracy (%)	F1 score (%)	Time (s)	Accuracy (%)	F1 score (%)	Time (s)	Accuracy (%)	F1 score (%)	Time (s)	Accuracy (%)	F1 score (%)	Time (s)
1	73.64	45.65	0.30	81.90	68.31	0.29	62.74	0	0.44	82.07	68.32	0.39
2	86.12	77.36	0.54	86.64	78.41	0.53	62.74	0	0.77	87.17	79.44	0.71
3	86.82	78.76	0.86	89.98	84.80	0.77	62.74	0	1.03	89.63	83.84	0.97
4	90.33	85.26	1.06	91.21	86.91	0.98	62.74	0	1.34	90.88	86.07	1.27
5	97.01	95.84	1.32	96.66	95.38	1.20	62.74	0	1.63	90.86	86.10	1.56
10	98.59	98.08	2.61	98.59	98.10	2.38	62.74	0	3.16	99.12	98.81	3.02
50	98.77	98.33	12.40	98.77	98.33	11.32	74.34	47.48	15.79	99.30	99.05	14.05
100	98.95	98.57	24.73	98.77	98.33	23.83	91.21	86.70	32.89	99.47	99.29	29.33
500	98.77	98.33	134.37	98.95	98.57	121.54	99.12	98.81	185.76	99.30	99.05	152.37

Result and discussion: In Table 4, these two models have similar performance in long-time training in original dataset. At epoch 500, the accuracy and F1 score for normal ANN is 98.77% and 98.33%. The accuracy and F1 score for second model is 98.95% and 98.57%. However, the model that contains GA-based feature selection shows a better performance in small epoch training. For instance, in epoch 1, the accuracy and F1 score of normal ANN is 73.64% and 45.65%, the accuracy and F1 score for second model is 81.90% and 68.31%. With GA-based feature selection, the accuracy improves 11%, the F1 score improves 50%. In noised dataset, the GA-based feature selection shows more evident improvement in limited training time. In long-time training, the accuracy and F1 score are still similar in two models. But in short time, the performance of the model with GA-based feature selection undeniably overcomes the normal ANN. For instance, in epoch 10, the accuracy and F1 score for normal ANN is just 62.74% and 0, the accuracy and F-score for another model is 99.12% and 98.81%. The results in two different datasets indicate that GA-based feature selection can improve the performance of classifier in limited time. And it can highly improve the performance of classifier when the dataset contains large number of noisy features. As for the reason, I think that GA-based feature selection can reduce the number of redundant and noisy features. At the same time, useful and discriminative features are kept. Therefore, with the optimal subset of features, the classifier performs better in limited training time. What is more, the GA-based feature selection is more powerful in dataset with large feature number. Since the dataset with larger feature number has more noisy features, which will obstruct the classifier to predict an accurate result.

The Table 4 also shows the GA-based feature selection can reduce the computation cost for same training epoch. In original dataset, the time cost for normal ANN at epoch 500 is 134.37s, and the time cost for second model is 121.54s at epoch 500. With GA-based feature selection, the time cost is 9.5% less. The result in noised dataset is more evident. The time cost for normal ANN at epoch 500 is 185.76s, and the time cost for second is 152.37s at epoch 500. The time cost decrease 21.9% with GA-based feature selection. As for the reason, since the GA-based feature selection can select the best feature subset. The number of input units also decrease. Thus, the ANN need less training time for same epoch.

3.4 Comparison between shared weights autoencoder and genetic algorithm-based feature selection

Motivation: The shared weights autoencoder has the potential risk of ignoring the discriminative features and decreasing the classifier performance [24]. Hopefully, GA-based feature selection can solve this problem by selecting informative features. This experiment is to compare the performance of shared weights autoencoder and GA-based feature selection.
Approach: To compare the impact of these two methods, two models are compared. One contains shared weights autoencoder and ANN, the other one contains GA-based feature selection and ANN. The training epoch for autoencoder is 500. For GA-based feature selection, the crossover rate and the mutation rate are same as previous experiments. Besides, the training epoch for classification ANN is 300. The average accuracy and F1 score in 10 times for these two models are compared. I also compare the average time cost in 10 times for training shared weights autoencoder and GA-based feature selection. The used computer is same as the one in previous experiments.

Table 5. The confusion matrix for ANN with GA-based feature selection

	Malignant (Confirmed)	Benign (Confirmed)	Total

Malignant (Predicted)	206	0	206
Benign (Predicted)	6	357	363
Total	212	357	569

Table 6. The confusion matrix for ANN with shared weights autoencoder

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

Table 7. The time cost for training shared weights autoencoder and GA-based feature selection

	Time cost (s)
Shared weights Autoencoder training	0.20
GA-based feature selection	309.51

Result and discussion: In Table 5 and 6, we can see that the accuracy and F1 score of ANN with GA-based feature selection is higher than ANN with autoencoder. The accuracy and F1 score for ANN with GA-based feature selection is 98.95% and 98.57%. The accuracy and F1 score for ANN with shared weights autoencoder is 97.54% and 96.67%. This shows the GA-based feature selection is better than shared weights autoencoder in improving the classifier performance. The reason may be as following. The main idea of shared weights autoencoder is to compress the original features into less features. In this process, redundant and noisy features can be ignored or deleted. However, some discriminative features may also be ignored. GA-based feature selection can solve this problem. By genetic algorithm, the best feature subset will be selected, noisy features are deleted, useful features are kept.

On the other hand, from Table 7, we can see that the time cost for GA-based feature selection is much larger than shared weights autoencoder. The time cost for shared weights autoencoder is just 0.2s, but the time cost for GA-based feature selection is 309.51s. The reason is as following. For shared weights autoencoder, we just need use backpropagation to train one model. However, in GA-based feature selection, in each generation, each individual represents one model, and every model need training. Therefore, the time cost for GA-based feature selection is much larger.

3.5 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 compare the performance of my two models with this baseline.

Approach: To compare the performance, I compare the average accuracy and F1 score of my models in 10 times with the logistic regression method results from W. H. Wolberg et al [1]. The training setting is same as previous experiments.

Table 8. 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

Result and discussion: From Table 5, 6 and 8, we can see that the accuracy and F1 score of my two models are much higher than the logistic regression. The accuracy and F1 score of the model with autoencoder are 97.54% and 96.67%. The accuracy and F1 score for the model with feature selection are 98.95% and 98.57%. And the accuracy and F1 score for logistic regression is 96.2% and 94.91%. That shows my two models both perform better than the logistic regression model. The high performance of my models may be related to the following 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 [21]. Secondly, logistic regression cannot avoid the negative influence of redundant features and noisy data. Autoencoder can solve this problem by feature extraction. Furthermore, the shared weights technique can improve the feature extraction ability for autoencoder. The GA-based feature selection can not only reduce the useless features but also keep the discriminative features. These selected features can improve the classification performance. Therefore, my two models should perform better than the logistic regression model.

4 Conclusion and Future Work

This paper represents two novel neural networks. One uses shared weights autoencoder to effectively extract important features. The other one uses GA-based feature selection to select discriminative and informative features. With

autoencoder, the classifier has superior performance in limited training time. Because autoencoder has the ability of extracting important features and ignoring noisy data. Furthermore, with shared weights technique, the autoencoder appears more robust and the feature extraction ability is highly improved. The GA-based feature selection not only helps classifier performs better in limited training time, but also reduces computation resources. And it has better performance than shared weights but cost more training time. What is more, my two models both show 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 techniques, the network shows better performance [13]. And I will try to combine the GA-based feature selection and traditional machine learning method. For example, I can use GA-based feature selection to get important features then use support vector machine 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.

[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 Propagation for 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. 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.

[18] 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.

[19] 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.

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

[21] 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

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

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

[24] P. Alirezazadeh, A. Fathi and F. Abdali-Mohammadi, "A Genetic Algorithm-Based Feature Selection for Kinship Verification," IEEE Signal Processing Letters, vol. 22, (12), pp. 2459-2463, 2015.

[25] C. F. Tsai, W. Eberle, and C. Y. Chu, “Genetic algorithms in feature and instance selection,” Knowl.-Based Syst., vol. 39, pp. 240–247, 2013.

[26] W. Siedlecki and J. Sklansky, "A note on genetic algorithms for large-scale feature selection," Pattern Recognition Letters, vol. 10, (5), pp. 335-347, 1989.

[27] Te-Sheng Li, ― Feature Selection For Classificatin By Using a GABased Neural Network Approach‖, Journal of the Chinese Institute of Industrial Engineers, Vol. 23, No. 1, pp. 55-64, 2006.

[28] C. E. Taylor, "Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Complex Adaptive Systems.John H. Holland," The Quarterly Review of Biology, vol. 69, (1), pp. 88-89, 1994.

[29] S. Aalaei et al, "Feature selection using genetic algorithm for breast cancer diagnosis: experiment on three different datasets," Iranian Journal of Basic Medical Sciences, vol. 19, (5), pp. 476-482, 2016.

[30] F.Z. Brill, D.E. Brown, and W.N. Martin, “Fast Genetic Selection of Features for Neural Network Classifiers,” IEEE Trans. Neural Networks, vol. 3, no. 2, pp. 324-328, Mar. 1992.

[31] J.H. Yang and V. Honavar, “Feature Subset Selection Using a Genetic Algorithm,” IEEE Intelligent Systems, vol. 13, no. 2, pp. 44- 49, 1998.

[32] L.I. Kuncheva and L.C. Jain, “Nearest Neighbor Classifier: Simultaneous Editing and Feature Selection,” Pattern Recognition Letters, vol. 20, pp. 1149-1156, 1999.