A simplified way of classifying breast cancer tumors

Bikram Kumar De
PhD Student
Institute for Software Integrated Systems
Vanderbilt University
Course: CS 5891 - Deep Learning

Abstract— **In this project, I am performing a task on determining breast tumor as malignant or benign. With the increasing number of breast cancers, it is very important and helpful to determine whether a tumor is malignant or benign. There has been works in this area previously which have achieved significantly high accuracy, but those works are quite complicated. Moreover, those works lacks a judgement of other available performance metrics. In this project, I have used a simple method of classifying breast cancer tumors using deep neural networks. This method is easy to understand as well as implement. I have achieved an accuracy of 98.83% on test data and 100% on training. Moreover, I have also analyzed the importance of other performance metrics like precision and recall and highlighted their significance in medical diagnosis.**

I Introduction

Breast cancer is cancer that develops from breast tissue. Signs of breast cancer may include a lump in the breast, a change in breast shape, dimpling of the skin, fluid coming from the nipple, a newly inverted nipple, or a red or scaly patch of skin.[1] The high incidence of breast cancer in women has increased significantly in the recent years. It is the cause of the most common cancer death in women (exceeded only by lung cancer). The use of classifier systems in medical diagnosis is increasing gradually. There is no doubt that evaluation of data taken from patients and decisions of experts are the most important factors in diagnosis. However, expert systems and different artificial intelligence techniques for classification also help doctors in a great deal. Classification systems can help in minimizing possible errors that can happen because of inexperienced experts and provide medical data to be examined in shorter time and in a more detailed manner.

II The problem

I would like to predict the presence of breast cancer by classifying the features from an image as benign or malignant. I am performing a supervised learning. It is essentially a classification task where I learn the data using a training set and then use the trained model to predict on a test set. For this task, I have selected the Wisconsin Breast Cancer data which was available in the UCI Machine Learning Repository.[2] Various features are obtained from the image which is stored in the dataset. Corresponding to those features, the label of tumor being malignant or benign is provided. Thus the entire task is converted to a binary classification with malignant as the positive labels and benign the negative ones.

III Related Work

This is a highly popular dataset and has been used multiple times and by various researchers in their work. Most of them have obtained a descent classification accuracy. The accuracy obtained by Pena-Reyes & Sipper (1999) [5] was 97.36% using fuzzy-GA method. Akay (2009) [3] combined SVM with feature selection obtaining highest classification accuracy (99.51%) for SVM model that contains five features. In Setiono (2000) [6], the classification was based on a feed forward neural network rule extraction algorithm. The reported accuracy was 98.10%. (Quinlan, 1996) [7] reached 94.74% classification accuracy using 10-fold cross-validation with C4.5 decision tree method. (Hamilton, Shan, & Cercone, 1996) [8] obtained 94.99% accuracy with RIAC method, while (Ster & Dobnikar, 1996) [9] obtained 96.8%

with linear discreet analysis method. The accuracy obtained by Nauck and Kruse (1999) [10] was 95.06% with neuron-fuzzy techniques.

IV Dataset Description

The dataset contains features that are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. The attribute information of the dataset is listed as follows. The 1st column contains a patient ID number. The next column contains the labels whether the tumor is malignant (=M) or benign (=B). For the columns 3 till 32 ten real-valued features are computed for each cell nucleus. They are described below as follows:

1) radius (mean of distances from center to points on the perimeter)
2) texture (standard deviation of gray scale values)
3) perimeter
4) area
5) smoothness(local variation in radius lengths)
6) compactness (perimeter2 / area - 1.0)
7) concavity (severity of concave portions of the contour)
8) concave points (number of concave portions of the contour)
9) symmetry
10) fractal dimension (”coastline approximation” - 1)

The mean, standard error, and ”worst” or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius. All the feature values are recorded with four significant digits. Of the 569 data points, 357 are benign and 212 are malignant. Also, it is worthwhile to mention that there are no missing values.

V My Idea

In the following section, I have discussed my thoughts on how to proceed to solve the problem. I have discussed the challenges faced as well as my way through those challenges. Then I have given a brief description on the implementation process of my algorithm. I have discussed multiple implementation procedures and stated the advantages and disadvantages of those.

V-A Preprocessing

The data set size is 569 x 33. Initially preprocessing is necessary. The first column is removed as it contains Id of the patient which provides us with no valuable information. Then the next column contains the labels. The malignant labels represented by M are changed to 1 and the benign labels represented by B are changed to 0. Thus, this essentially becomes a binary classification.

Now, observing the rest of the columns with the values provided, I found that the data lies in varying ranges. For example, texture standard error (texture_se) has very small values mostly below 1 whereas, area standard error (area_se) has large values near 100. These types of varying values would create issues with my deep neural network. Thus, there is a need for feature scaling for normalization of the data.

For this dataset, I used minmax normalization. Minmax normalization is a normalization strategy which linearly transforms x to y = (x-min)/(max-min), where min and max are the minimum and maximum values in X, where X is the set of observed values of x. This method results in rescaling the range of features to the scale in range [0, 1]. Next, I divided the data into training dataset and test dataset. I shuffled the data and randomly selected 85% of the data for training and remaining 15% for testing.

V-B Network Model — Deep Feed Forward Network

I decided to use deep feedforward network for the dataset. I built the model from scratch. I performed a vectorized implementation using the standard Numpy library. This helped speed up the execution process. Moreover, I used a modular form of writing the code so that functions can be reused repeatedly. The utils file contains all the various functions which are necessary, and it has been imported in the jupyter notebook. It contains functions for the forward propagation step, the backward propagation step, the cost function as well as functions for printing the results and plotting the cost function as a function of the number of epochs. The equations of forward propagations are as follows:

Z = $W^T X + b$.

A = σ(Z), where σ is the activation function. W and b are the weights and the biases.

V-C Network Model — Feed Forward Network Using Keras

The deep feed forward network developed from scratch is very helpful and the accuracy received after implementing a simple model is noteworthy. However, the implementation process is long, and it results in a long code consisting of huge number of lines.

To avoid these disadvantages, I decided to try out a standard model from Keras. I used the Keras Sequential Model. I added a Dense layer for each layer of the network. I used ReLU activation function or the TanH in the hidden layers. Sigmoid is used in the output layer since it is a binary classification.

While compiling, I have used binary cross entropy as my loss function. While evaluating the model, the standard Keras accuracy is printed as the score.

VI Experiments and Results

Since, it is a binary classification, I have used sigmoid activation function in the final output layers. For the hidden layers, I have experimented with two different activation function: Rectified Linear Unit (ReLU) and TanH.

VI-A Deep Feed Forward Network

I have initially experimented with unregularized network. Then I used two different regularization techniques: L2 Regularization and Dropout. After that I tried optimizing my network using Adam optimizer. Apart from that, I have varied my hyperparameters like learning rate, lambda, beta1 and beta2 to check the best possible result. I have shown all the results in a tabular format.

I have taken two cases: (a) I have used ReLU as my activation function and used a 30x20x10x1 network. I have represented the results in Table 1. (b) I have used tanH as my activation function and used a 30x50x20x5x1 network. These results are represented in Table 2.

I have also plotted the cost functions vs the no. of epochs. Figure 1 shows the cost functions for a neural network with 4 layers and ReLU activation function in the hidden layer. Figure 2 shows the cost functions for neural network with 5 layers with tanh as the activation function for the hidden layers.

VI-B Feedforward Network with Keras

Using Keras, I have essentially varied my optimizers to check the result. I have used Stochastic Gradient Descent, Ada Delta, RMS Prop and Adam and checked the results. Along with this I have used L2 Regularization and Dropout. The code is written in a modular format. The functions are called from inside a loop such that the minimum lines of code can perform the maximum amount of work.

I have experimented with two different networks: (a) A 30x20x5x1 network with tanH as activation function. The results are displayed in Table 3. (b) A 30x64x32x16x4x1 network with ReLU activation function. The results are displayed in Table 4.

VII Analysis

I have shown the accuracy, precision, recall and f score for various cases involving changing the architecture, and the hyperparameters. I have also used regularization methods like L2 and dropout. Finally, I optimized the network trying various optimizer like SGD, Ada Delta, Adam and RMS Prop.

VII-A Feed Forward Network

In my feed forward network, I observed that regularization has shown an improvement over unregularized neural network. The value of precision and recall also increases significantly on regularization. However high precision obtained in dropout with low recall is useless. It means that the model classified most as benign. It is useless because the model is missing out on lots of actual cases. We would need a high precision along with a high recall for our model to be effective in detecting tumors.

TABLE 1: ReLU with 30x20x10x1 network

Activation - ReLU Net: 30x20x10x1		Accuracy		Precision		Recall		F - Score	
		Training	Testing	Training	Testing	Training	Testing	Training	Testing
Unregularized	α = 0.001	90.68%	81.39%	97.14%	100%	76.83%	54.28%	85.8%	70.37%
	α = 0.1	93.37%	90.69%	90.05%	88.57%	92.09%	88.57%	91.06%	88.57%
	α = 0.8	93.99%	89.53%	88.54%	86.11%	96.04%	88.57%	92.14%	87.32%
L2 Regularized	λ = 0.3	92.54%	89.53%	93.25%	93.33%	85.87%	80%	89.41%	86.15%
	λ = 0.6	92.75%	89.53%	93.82%	93.33%	85.87%	80%	89.67%	86.15%
Dropout	ρ = 0.5	92.96%	89.53%	99.31%	100%	81.35%	74.28%	89.44%	85.24%
	ρ = 0.1	70.6%	63.95%	100%	100%	19.77%	11.42%	33.01%	20.51%
Adam Optimized	β1 = 0.9 β2 = 0.9	96.42%	94.18%	100%	94.11%	88.88%	91.42%	94.11%	92.75%
	β1 = 0.9 β2 = 0.6	100%	95.34%	100%	96.96%	100%	91.42%	100%	94.11%

TABLE 2: TanH with 30x50x20x5x1 network

Activation - TanH Net: 30x50x20x5x1		Accuracy		Precision		Recall		F- Score	
		Training	Testing	Training	Testing	Training	Testing	Training	Testing
Unregularized	α = 0.001	90.47%	91.86%	92.07%	88.88%	82.06%	85.71%	86.78%	87.27%
	α = 0.1	93.16%	94.18%	98.08%	100%	83.69%	82.14%	90.32%	90.19%
	α = 0.8	94.4%	93.02%	91.97%	84.37%	93.47%	96.42%	92.72%	89.99%
L2 Regularized	λ = 0.3	93.58%	93.02%	94.73%	86.66%	88.04%	92.85%	91.26%	89.65%
	λ = 0.9	93.37%	91.86%	94.7%	86.2%	87.5%	89.28%	90.96%	87.71%
Dropout	ρ = 0.7	92.75%	94.18%	98.06%	100%	82.6%	82.14%	89.67%	90.19%
	ρ = 0.3	87.57%	91.86%	100%	100%	67.39%	75%	80.51%	85.71%
Adam Optimized	β1 = 0.9 β2 = 0.9	100%	97.67%	100%	100%	100%	92.85%	100%	96.29%
	β1 = 0.9 β2 = 0.6	100%	97.67%	100%	93.33%	100%	100%	100%	96.55%
	β1 = 0.1 β2 = 0.01	100%	**98.83%**	100%	96.55%	100%	100%	100%	98.24%

Along with this, we can note that there is significant overfitting of the data. In many cases, we obtain a high training accuracy with a lower testing accuracy. This mainly occurs since our dataset size is low. However, increasing the number of layers shows a reduction in overfitting. I also observed that optimization of our model has helped the performance significantly. There is a reduction in overfitting. We obtained a high recall (**100%**) which means we have been able to correctly identify all the cases. The precision in that case was almost **97%**. Along with that we obtain an accuracy of **98.83%**. So, we can safely say that increasing the layers along with optimizing the network has helped the performance metrics like accuracy, recall significantly. The above analysis can also be verified by watching the plotted cost functions.

VII-B Feedforward Network using Keras

While developing a neural network from scratch helps in understand the underlying dynamics and mathematics in the functioning of the network, a Keras model on the other hand works much simpler and gives the result much faster. Here, I have used four different optimizers along with

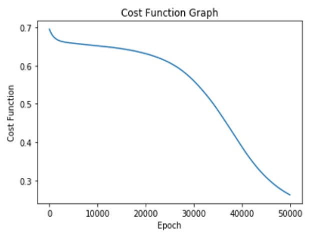

Figure 1a: Unregularized

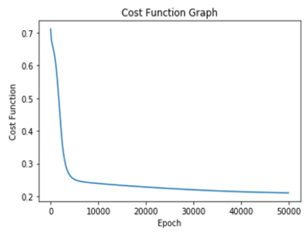

Figure 1b: L2 Regularized

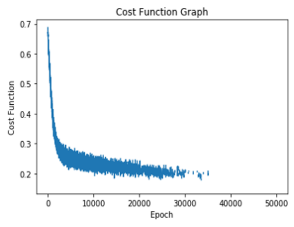

Figure 1c: Dropout

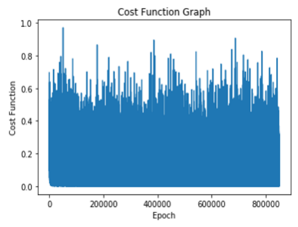

Figure 1d: Adam Optimizer

Figure 1: Cost Functions for ReLU with 30x20x10x1 Network

TABLE 3: Various Results for TanH activation with 30x20x5x1 Network

Optimizers	Regularizers		Training Accuracy	Testing Accuracy
SGD	Dropout	0.5	92.75%	89.53%
		0.3	92.13%	87.2%
	L2		90.68%	84.88%
	None		92.54%	90.69%
AdaDelta	Dropout	0.5	94.2%	94.18%
		0.3	93.37%	97.67%
	L2		92.33%	88.37%
	None		93.99%	91.86%
RMS Prop	Dropout	0.5	93.99%	96.51%
		0.3	94.82%	96.51%
	L2		92.54%	90.69%
	None		94.82%	96.51%
Adam	Dropout	0.5	93.99%	93.02%
		0.3	94.82%	96.51%
	L2		92.96%	91.86%
	None		95.23%	95.34%

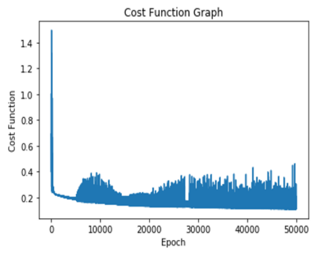

Figure 2a: Unregularized

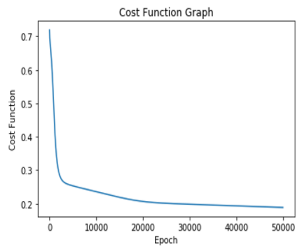

Figure 2b: L2 Regularized

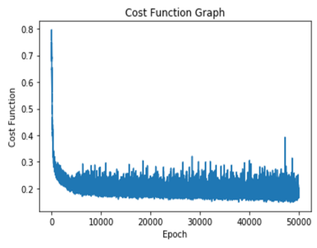

Figure 2c: Dropout

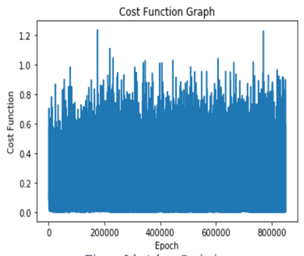

Figure 2d: Adam Optimizer

Figure 2: Cost Functions for TanH with 30x50x20x5x1 Network

TABLE 4: Various Results for ReLU activation with 30x64x32x16x4x1 Network

Optimizers	Regularizers		Training Accuracy	Testing Accuracy
SGD	Dropout	0.5	90.47%	94.18%
		0.3	90.89%	95.34%
	L2		61.07%	72.09%
	None		91.71%	94.18%
AdaDelta	Dropout	0.5	93.99%	95.34%
		0.3	94.2%	96.51%
	L2		61.07%	72.09%
	None		94.82%	94.18%
RMS Prop	Dropout	0.5	93.37%	96.51%
		0.3	94.4%	96.51%
	L2		61.07%	72.09%
	None		95.03%	**98.83%**
Adam	Dropout	0.5	93.58%	95.34%
		0.3	95.03%	94.18%
	L2		61.07%	72.09%
	None		95.44%	94.18%

regularization techniques. It is observed that L2 regularization with optimization gives a poor performance. Among all the optimizers, RMS Prop performs the best with the accuracy of **98.83%**. Other than that, Ada Delta and Adam also shows high performance. SGD with Nesterov momentum essentially lags behind other optimizers. It is worthwhile to note that, overfitting of data has almost been removed with test accuracy greater than training in multiple occasions and the difference between them is insignificant mostly.

VIII Conclusion

I have shown a simple method for classifying breast cancer tumors. The accuracy obtained is among the best and is a competitive one. Moreover, the previous methods used are much more complex involving complicated machine learning algorithms which are difficult to implement as well as time consuming. The previous best obtained was using SVM combining with feature selection. Compared to that, I have used simple neural network techniques to obtain almost similar accuracy.

Other methods which have obtained lesser accuracy are also complicated some involving Genetic Algorithms. I have shown a comparative study between my work and previous works in Table 5.

Another thing to note is that, most work have used only accuracy as the metrics. However, considering the type of data and the task that is performed, other metrics like precision and recall can give much better measure of the performance of our model. So, I have considered them along with the balance between precision and recall i.e. f score which is the harmonic mean between the two.

Table 5: Comparison with existing works

Paper	Method	Accuracy
Pena-Reyes & Sipper (1999) [5]	Fuzzy-GA	97.36%
Akay (2009) [3]	SVM with feature selection	**99.51%**
Setiono (2000) [6]	Neural Net Rule extraction	98.10%
Quinlan, 1996) [7]	10-fold cross-validation with C4.5 decision tree	94.74%
Hamilton, Shan, & Cercone, 1996 [8]	RIAC	94.99%
Ster & Dobnikar, 1996 [9]	Linear discreet analysis	96.8%
Nauck and Kruse (1999) [10]	Neuron-fuzzy	95.06%
My Project	Neural Network with Optimization	**98.83%** with **100%** Recall

This helped determine how many of the predicted tumors are true (precision) as well as how many of the actual tumors I could predict correctly (recall). Considering early detection of tumors is helpful in curing cancers, having high recall is very important. Along with that the precision should also be high which can be monitored by the f score.

IX Future Work

One of the disadvantages of the data set that can be improved in the future is the size of the data. It is a small data set consisting of only 569 samples. Thus, this forms a challenge in training and testing. Enough samples are not available for training if we want to have a reasonable size of test data set. The size of data forms a major issue in most machine learning algorithms. It affects neural network the most. In future, this data set can be extended by updating the latest cases of tumors.

Also, there are only 30 features in this data set. In future, further features might be incorporated in the data set to check their effect on the performance.

If these slight improvements to the data set can be made, I am sure the performance of my algorithm will be much better. Apart from these, I have used few different architectures for this project. There are lots of other possible architectures which can be explored in future along with other activation functions to check the performance of the model.

References

[1] Wikipedia article on breast cancer.

[2] Dataset UCI Machine Learning Repository: http://archive.ics.uci.edu/ml/datasets/BreastCancerWisconsin

[3] Akay, M. F. (2009). Support vector machines combined with feature selection for breast cancer diagnosis. Expert systems with applications, 36(2), 3240-3247.

[4] Abdel-Zaher, Ahmed M., and Ayman M. Eldeib. ”Breast cancer classification using deep belief networks.” Expert Systems with Applications 46 (2016): 139-144.

[5] Pena-Reyes, C. A., & Sipper, M. (1999). A fuzzy-genetic approach to breast cancer diagnosis. Artificial Intelligence in Medicine (17), 131155.

[6] Setiono, Rudy. ”Generating concise and accurate classification rules for breast cancer diagnosis.” Artificial Intelligence in medicine 18, no. 3 (2000): 205-219.

[7] Quinlan, J. R. (1996). Improved use of continuous attributes in C4.5. Journal of Artificial Intelligence Research, 4, 7790.

[8] Hamilton, Howard John, Nick Cercone, and Ning Shan. RIAC: a rule induction algorithm based on approximate classification. Computer Science Department, University of Regina, 1996.

[9] Ster, Branko, and Andrej Dobnikar. ”Neural networks in medical diagnosis: Comparison with other methods.” In International conference on engineering applications of neural networks, pp. 427-30. 1996.

[10] Nauck, D., & Kruse, R. (1999). Obtaining interpretable fuzzy classification rules from medical data. Artificial Intelligence in Medicine, 16, 149169.