Wisconsin Breast Cancer Diagnosis Deep Learning Revisited

By

Yuefeng Zhang

November 2017

1. Introduction

The machine learning methodology has long been used in medical diagnosis [1]. The Wisconsin Breast Cancer Database (WBCD) dataset [2] has been widely used in research experiments.

Most of publications focused on traditional machine learning methods such as decision trees and decision tree-based ensemble methods [5].

Recently supervised deep learning method starts to get attention. For instance, Stahl [3] and Geekette [4] applied this method to the WBCD dataset [2] for breast cancer diagnosis using feature values calculated from digitized image of a Fine Needle Aspirate (FNA) of a breast mass.
These features describe the characteristics of the cell nuclei present in the image.

Given the list of features calculated from a digitized image of the FNA of a breast mass from a patient, the problem is how to diagnose (determine) whether or not the patient has breast cancer. This problem can be treated as a 2-class (*benign* or *malignant*) classification problem.

Stahl [3] used the WBCD dataset with derived features (e.g., mean, standard error, …, etc.) and experimented three types of deep neuron networks: 1, 2, and 3 hidden layers of 30 neurons without any data pre-processing. Geekette [4] used only the originally identified features with pre-processing such as center and scale, but did not provide the details of the neuron network architecture in use.

This capstone project report presents a new supervised deep learning method for analyzing the same WBCD dataset [2] for breast cancer diagnosis using common open source libraries. The new supervised deep learning method inherits the merits of the methods experimented by Stahl [3] and Geekette [4]. Specifically, similarly to [4], the new method uses the originally identified features [2], pre-processes data using center and scale, and treats the breast cancer diagnosis problem as a 2-class classification problem. Like [3], the new method adopts multiple (four in this capstone project) hidden layers of deep neuron network.

The rest of this report consists of 4 sections. Section 2 discusses the dataset, and related issues and pre-processing solutions. Then Section 3 describes the new supervised deep learning architecture. After that, Section 4 discusses the evaluation results of applying the new deep learning network to the WBCD dataset. Finally Section 5 is the conclusion.

2. Dataset and Dataset Pre-Processing

The dataset used in this project is the publically available WBCD dataset [2]: https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data

As described in the WBCD dataset [1], the following features were computed for each cell nucleus and will be used as inputs to machine learning model:

- Clump Thickness
- Uniformity of Cell Size
- Uniformity of Cell Shape
- Marginal Adhesion
- Single Epithelial Cell Size
- Bare Nuclei
- Bland Chromatin
- Normal Nucleoli
- Mitoses

The following issues were observed with the dataset:

- Missing data
- Small dataset size
- Various ranges of data values
- Skewed data

2.1 Missing Data

One of the common issues with dataset is missing data and there is no exception to the WBCD dataset. Specifically quite a few Bare Nuclei entries are missing (marked as ?). There missing data entries are replaced with 0 for simplicity in this capstone project.

2.2 Small Dataset Size

The original WBCD dataset [2] consists of only 699 samples, which is too small for deep learning after dividing the dataset into training and testing subsets.

This small data size issue is resolved in this capstone project by generating new data samples as follows. First the feature columns are separated from the dataset by dropping the ID and Diagnosis/Class columns. Then the Numpy random number generation library *Numpy.random.normal*(*mean, sigma, features.shap*e) is used to generate an array of normal distributed random numbers with the same feature dimensions as the given dataset, where *mean* = 0 and the standard deviation *sigma* = 0.1. After that, the generated array of random numbers are added into the given dataset element by element to form a new features dataset.

The last column "Diagnosis/Class" in the WBCD dataset [2] is used as label in training deep learning model in this capstone project. These labels (i.e., the values of the Diagnosis or Class column) are reused to label the corresponding samples (feature vectors) in the new generated dataset. This is achieved by combining the new generated array of features with the "Diagnosis/Class" column from the given dataset into a new dataset.

This dataset generation process is repeated to generate multiple new datasets. Finally all of the new datasets are combined with the original dataset to form a final dataset of 30,756 samples in total for experiment in this capstone project.

2.3 Various Ranges of Data Values

The ranges of feature values for different features are different. As a common practice, for each of the features, the feature values are scaled into the range of [0, 1] for deep learning as follows:

$$(Value - Minimum) / (Maximum - Minimum)$$

2.4 Skewed Data

Figure 1 shows the Pandas generated scatter plot matrix of the dataset after data pre-processing as described in Sections 2.1, 2.2, and 2.3. It can be seen from the figure that the dataset is skewed to the right.

This data skewness issue is significantly alleviated by taking squired root of the feature values. Figure 2 shows the result of applying squired root transformation to the dataset.

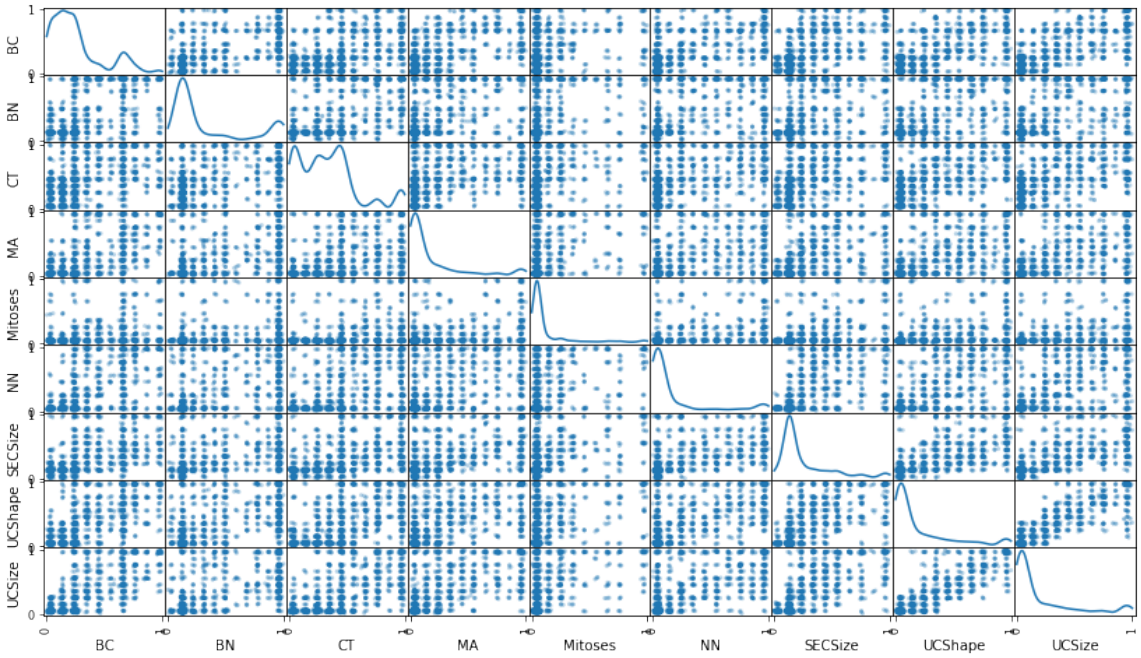

Figure 1: Skewed dataset.

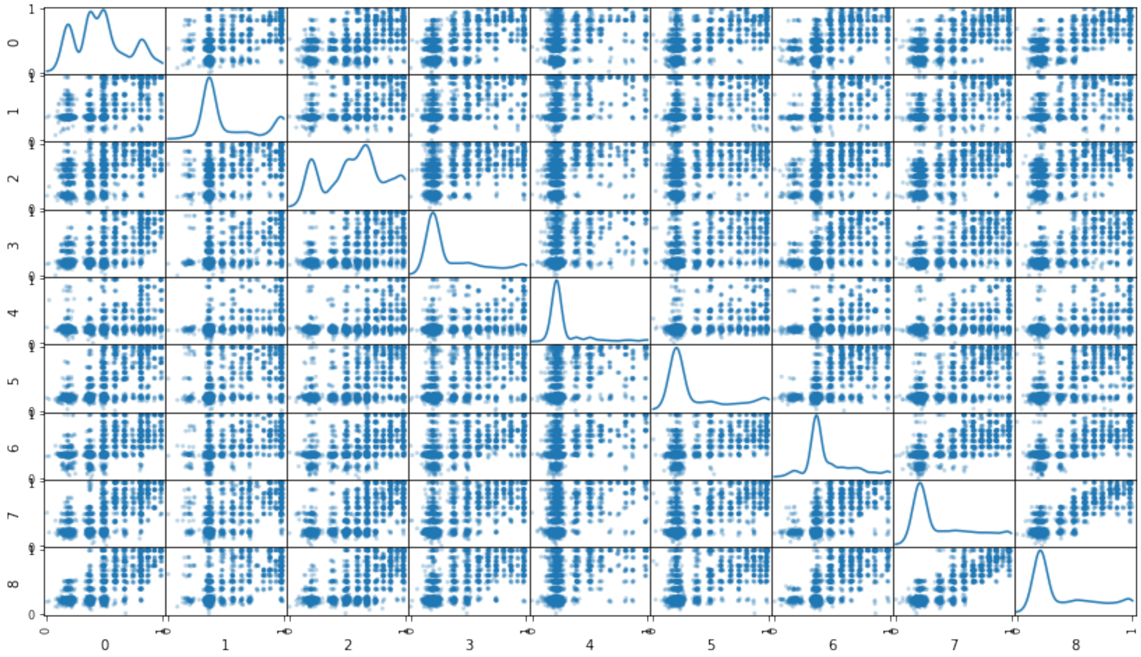

Figure 2: Transformed dataset.

3. New Deep Learning Network

As shown in Figure 3, similarly to [3], four hidden layers are used in the new deep learning network for the capstone project. In addition, dropout layers with dropout rate of 0.5% are introduced between hidden layers to avoid overfitting and improve the new deep learning network performance in accuracy.

As described before, the inputs to the new deep learning network are the feature values computed from digitized image of FNA, and the outputs from the network are the two identified classification diagnosis classes B (*benign*) or M (*malignant*).

The first hidden layer uses 9 neurons and the last two hidden layers use 1 neuron to match the number of input features and the number of output classes respectively. The size (5 neurons) of the hidden layer in the middle is in the middle of the sizes of the first and last two hidden layers.

The common rectified linear unit (ReLU) activation function *relu* is used in hidden layers 1, 2, and 3, while the widely used *sigmoid* activation function is used in the final hidden layer 4 to produce continuous output in the range of (0, 1). A threshold of 0.5 is inherently used to generate binary diagnosis output from the continuous output.

The following loss function, optimizer, and metrics functions are used in training the new deep learning network:

- **Loss function:** binary cross entropy
- **Optimizer function:** Adam (Adaptive Moment Estimation)
- **Metrics function:** accuracy

The popular open source Keras deep learning library [7] is utilized for the implementation of this new deep learning network in this capstone project.

More implementation details of the new deep learning network are available in GitHub [8].

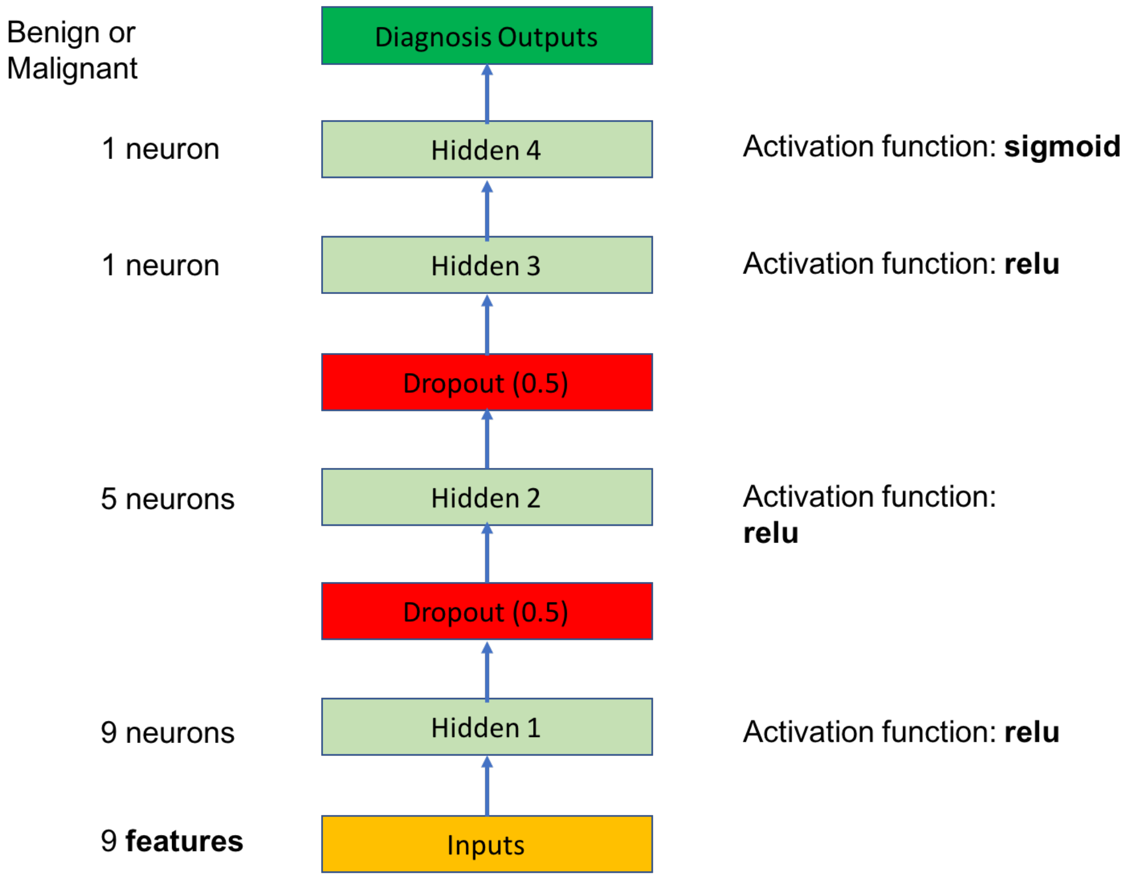

Figure 3: The architecture of the new deep neuron network.

4. Evaluation Results

Similarly to [4], the new dataset is split into two parts: 75% for training and the remaining 25% for testing. The major difference between the new dataset used in this capstone project and the original dataset used in [4] is that the new dataset of 30,756 samples is 44X bigger than the original dataset of 699 samples.

The result of the new supervisor deep learning method is compared with the result of applying the open source scikin-learn out-of-box implementation of the Random Forest Classifier [6] with default settings to measure the relative performance.

Similar to the evaluation method practiced by Geekette [4], The accuracy is used as the evaluation metrics.

The table below shows the summary of the comparison results of applying the testing dataset to the trained Random Forest Classifier and the new deep learning network models.

The following default settings were used in training and testing the sciki-learn out-of-box implementation of the Random Forest Classifier model:

- *n_estimators=10*,
- *criterion='gini'*,
- *max_depth=None*,
- *min_samples_split=2*,
- *min_samples_leaf=1*,
- *min_weight_fraction_leaf=0.0*,
- *max_features='auto'*,
- *max_leaf_nodes=None*,
- *min_impurity_decrease=0.0*,
- *min_impurity_split=None*,
- *bootstrap=True*,
- *oob_score=False*,
- *n_jobs=1*,
- *random_state=None*,
- *verbose=0*,
- *warm_start=False*,
- *class_weight=None*

The following are the settings of the new deep learning network:

- architecture: 4 hidden payers, 2 dropout layers with dropout rate of 0.5%, …, etc. (see Section 3 for details)
- batch size = 32
- 500 epochs

Machine Learning Algorithm	**Settings**	**Accuracy**
Scikin-learn Random Forest Classifier	Default (e.g., *n_estimators=10*, *criterion='gini'*, …, etc.)	0.988555

Keras Deep Learning Network	epochs = 500
	batch_size = 32
		0.982833

Table 1: Summary of model testing results.

As shown in the above table, the performance of the new deep learning network is competitive with the Random Forest Classifier in accuracy. Both of them achieved more than 98% in accuracy.

5. Conclusion

This capstone project presented a new deep learning network for the classification of the WBCD dataset [1][2].

In this capstone project the original dataset was extended 44 times by introducing normal distributed random noise into existing data samples. The missing data entries with the Bare Nuclei feature were replaced with zero and then the data values were scaled into the range of [0, 1] for deep learning. The skewness problem with the dataset was significantly alleviated by taking squired root of the feature values.

It was observed in the new deep learning model training and testing experiments that using deep learning (with 4 hidden layers) in conjunction with Dropout layers and Sigmoid activation function in the final hidden layer played a key role in improving deep learning performance in classification accuracy. The experimental results (see Section 4) demonstrated that by using deep learning (with 4 hidden layers) and introducing Dropout layers and Sigmoid activation function in the final hidden layer, the new deep learning network is competitive with the traditional Random Forest Classifier in the classification of the WBCD dataset [1][2].

The major advantage of deep learning network over the traditional machine learning models such as Random Forest Classifier is that it is much more flexible and capable in handling various complicated machine learning problems in many different domains. For example, in computer vision and object detection, each pixel of an image is a feature and thus the total number of features can easily be millions. This type of image classification problem cannot be handled by the traditional supervised machine learning models.

References

[1] W. H. Wolberg, etc., "Wisconsin Breast Cancer Database (WBCD)": https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.names , January 8, 1991

[2] WBCD data file: http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data, January 8, 1991

[3] K. Stahl, "Wisconsin Breast Cancer Diagnosis Deep Learning":
http://www.rpubs.com/kstahl/wdbc_ann, July 17, 2017

[4] D. Geekette, "Breast Cancer data: Machine Learning & Analysis":
http://rpubs.com/elena_petrova/breastcancer, November 25, 2016

[5] L. a comment, "Random forest classifier combined with feature selection for breast cancer diagnosis and prognostic", Scientific Research, Vol.6 No.5, May 2013, http://www.scirp.org/journal/PaperInformation.aspx?PaperID=31887

[6] sk-learn Random Forest Classifier: http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[7] Keras: The Python deep learning library: https://keras.io/

[8] Y. Zhang, Capstone Project in GitHub https://github.com/mlyuefeng/machine-learning/tree/master/capstone