Breast Cancer Detection Using Low-Computation-Based Collaborating Forward-Dependent Neural Networks

Karan Sanwal and Himanshu Ahuja

Abstract Considering the eminence of breast cancer detection, a variety of models have been proposed for its diagnosis. Although many of these models provide high-performance accuracy, they are computationally expensive. We propose a novel ensembling method, CFDNN which aims at reducing the computational expense, whilst maintaining the high-performance accuracy comparable to the state-of-the-art models in the breast cancer detection problem. To the best of our knowledge, the proposed CFDNN model is the computationally fastest model in breast cancer detection with an accuracy of 99.01%, making CFDNN an optimal choice for providing a second opinion in medical diagnosis.

Keywords Computer-aided diagnosis · Breast cancer detection · Neural network ensemble

1 Introduction

Cells are the smallest structural and functional units in any living being. Moreover, these cells have the capability to transfer their structural and functional characteristics into daughter cells. Cancer is caused by the abnormal cells, which divide uncontrollably and destroy the tissue of origin. This uncontrollable division of the cancer cells leads to the formation of a tumour. A tumour can be further classified into either benign or malignant. A benign tumour does not intrude on the surrounding tissue; hence, it is in a way controllable. Most benign tumours are rendered curable through modern medicinal techniques.

Around the world, breast cancer is the most prevalent cancer diagnosed in women. In the United States alone, an estimated 250,000 new patients and approximately 40,450 deaths were reported in 2016 [1]. These startling numbers remark the importance and the urgent need for early detection of breast cancer. In such a situation, machine-aided diagnosis can be utilised for a second opinion in medical.

K. Sanwal (✉) · H. Ahuja
Computer Science Department, Delhi Technological University, New Delhi, India
e-mail: karansanwal@gmail.com

M. L. Kolhe et al. (eds.), *Advances in Data and Information Sciences*, Lecture Notes in Networks and Systems 39, https://doi.org/10.1007/978-981-13-0277-0_9

It is also of utmost importance that the methods for training these models to be computationally fast, so their access becomes more diverse and extensible. Most algorithms today are computationally expensive. To the best our knowledge, the novel ensembling method, CFDNN, is the computationally least expensive approach in breast cancer diagnosis. CFDNN provides an accuracy of 99.01% in just two layers of neural network ensemble on Wisconsin Breast Cancer Dataset.

2 Related Work

Many algorithms have been proposed for breast cancer diagnosis using the Wisconsin Breast Cancer Dataset (WBCD) in literature [2–6], which is a two class-classification problem in machine learning.

The AMMLP algorithm proposed by Marcano-Cedeno et al. [7] achieved an accuracy of 99.36% using artificial meta-plasticity which prioritises update of weights for less frequent activations over the frequent ones. Abdel-Zaher and Eldeib [8] used a deep belief network for pretraining followed by a supervised phase utilising a neural network with backpropagation. Their model achieved a classification accuracy of 99.68%.

Devi and Devi [6] used a three-step procedure for early diagnosis of breast cancer, by treating the problem statement as that of outlier detection. In the first step, they clustered the data using the farthest-first algorithm, after which they applied an outlier detection algorithm. Finally, the J48 classification algorithm was used to classify the instances. Their model achieved an overall accuracy of 99.9% via the tenfold cross-validation technique.

Akay [3] used support vector machines combined with an F-score based feature selection method to achieve a classification accuracy of 99.51% using 80–20 data split. Nahato et al. [9] presented the algorithm RS-BPNN which first constructed rough sets to fill in the missing data and then used backpropagation algorithm to train the neural networks, to achieve a classification accuracy of 98.6%. Sayed et al. [10] proposed the use of a meta-heuristic optimization technique to chose features for classification, called Whale Optimization Algorithm (WOA) and implemented it on the WBCD.

Genetic algorithms have been extensively used to modify the structure and parameters of the neural networks and selectively chose the best model structure in ensemble modelling or to select the most significant features. Alickovic and Subasi [5] achieved 99.48% accuracy through the implementation of rotation forests for classification, and feature selection through genetic algorithms to eliminate the insignificant features. Bhardwaj and Tiwari [11] defined their crossover and mutation operations and utilised genetic algorithms to produce neural network offspring according to their fitness. It obtains an overall accuracy of 99.26%.

3 Wisconsin Breast Cancer Database

Developed by Dr. Willian H. Wolberg at the University of Wisconsin Hospital, the Wisconsin Breast Cancer Dataset contains a total of 699 training instances but due to missing features, 16 examples were omitted. The resultant dataset contains 444 training examples of benign tumour and 239 examples of a malignant tumour. Each training example consists of 10 features. The sample code feature is the id of each instance and is not considered during training or prediction. All the other attributes are integers from 1–10 [12].

4 Proposed Approach

The proposed algorithm, CFDNN, is aimed at reducing the computation associated with the machine-based detection of breast cancer. The algorithm leverages the following criteria for a faster performance:

1. Reducing computations at the base units.
2. Implementing a parallelizable global structure that can scale and take advantages of a multiprocessor-based architecture.

With the above being mentioned, the sensitivity of the problem at hand cannot be overlooked. CFDNN aims at achieving high accuracy at the lowest possible computational expense through a novel ensembling algorithm. Ensembling refers to a popular technique in machine learning, where, rather than using a single complex classifier, multiple weak classifiers are used before arriving at a single solution to the problem [5, 13–15]. However, most ensembling techniques in literature [2, 8] are computationally expensive due to the following reasons:

1. Large amount of computations wasted in increasing the base classifier accuracy.
2. Large amount of computations wasted in the creation of the ensemble in a way that introduces variation among its members.

CFDNN introduces an ensembling technique that intuitively creates varied classifiers which aim to reduce the error intersections amidst them. A property is far more salient than targeting a single classifier accuracy as a metric for the entire ensemble. The algorithm takes inspiration from the way students may effectively collaborate towards a group test. Each neural network in the ensemble is seen as a student, and the training set is seen as the syllabus of a test that the neural networks have to perform well on. Rather than the popular technique of dividing the syllabus by volume (Dividing the training set amidst the various neural networks), CFDNN divides it by difficulty. Each successive neural network studies only that part of the syllabus where its predecessor was unable to learn effectively.

Following this section, we introduce three computationally inexpensive flavours of the algorithm, based on the architecture of the system the model is running and the user requirements.

1. Serial CFDNN, the baseline CFDNN architecture
2. Parallel CFDNN Version: 1
3. Parallel CFDNN Version: 2

4.1 Serial CFDNN

Figure 1 highlights the architecture of the CFDNN model. The algorithm's base unit utilises a neural net with a single hidden layer comprising five nodes each. We define the *depth* of a neural network as a step-incremented variable (Assigned to *one*, for the first neural network) which describes the layer depth of the neural network in the ensemble model. The neural network created at *zero depth* is termed as the *Central Classifier (CF)* which is made to train on the entire dataset. The number of iterations (α) for learning is optimally set as per the dataset through k-cross-validation techniques. Due to the mentioned restrictions on its structure and training, when evaluated on the dataset, it is thence expected to misclassify a set of instances it could not learn. Once the *zero depth* neural network is trained on the entire dataset, a second neural network (with $depth = 1$) is initialised and is made to train only on the misclassifications of the first neural network. This pattern is repeated along the layer depth of the ensemble where the ith neural network trains solely on the misclassifications of the $(i-1)$th neural network. It is hence easy to intuit that the CF neural network has the most general idea of the dataset and the subsequent neural networks keep on specialising on the instances that are seen as an anomaly by the patterns learnt by the lower depth neural networks. It can be observed that the central

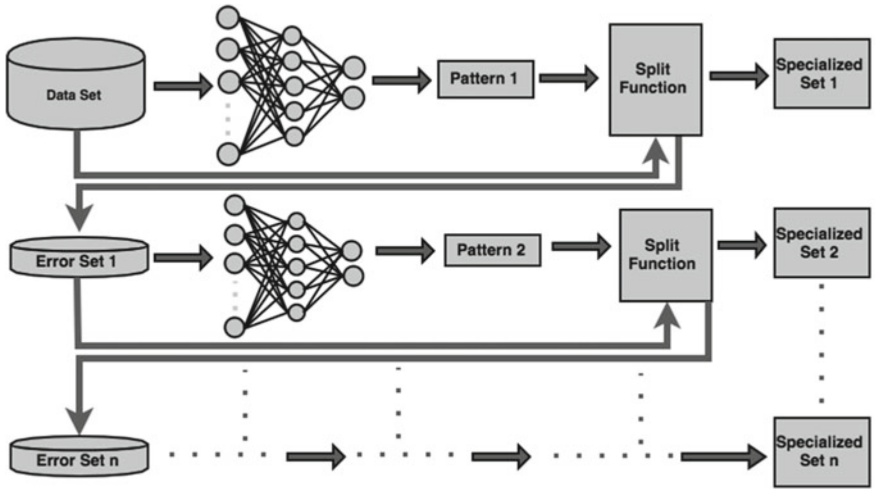

Fig. 1 The *n-depth* Serial CFDNN architecture

classifier splits the dataset into two sets: One which can be classified by the general pattern learnt by a weak-restricted CF and then another can be seen as an anomaly with respect to it. This relatively anomalous dataset is fed to the subsequent neural network, which, being a weak-restricted classifier itself, divides it into two sets in a similar fashion and the pattern is repeated along the depth. The general dataset division equation is given by

$$D_i = G_i \cup A_i \quad \text{and} \quad D_{i+1} = A_i \quad \forall i \in \{1, 2, 3, \ldots, n\}, \tag{1}$$

where n is the total depth of the neural network. G_i (General dataset for the ith neural network) contains the training examples in the dataset (D_i) in which the ith neural network correctly classifies and A_i (Anomalous dataset for the ith neural network) contains the training examples in which the ith neural network misclassifies.

This novel technique in ensembling is seen better for the following reasons:

1. To force the weak neural network to learn the anomaly dataset (A_i), the optimiser might shift its parameters in a way that it loses some accuracy over the general dataset.
2. Since the next neural network is only trained on the anomaly dataset (A_i), it is relatively specialised in that dataset compared to its predecessor neural network which is trained on the superset $G_i \cup A_i$. This property helps to create a low-computation-based mapper function which helps in increasing the effectiveness of parallelization as we will see in Sect. 4.2.
3. This also aids the idea of focussing on reducing the intersection between error sets rather than increasing sole accuracies of classifiers. It is easy to see that the higher depth neural networks will be weaker than the predecessor neural networks, but it will be able to solve instances that the predecessor neural network was unable to solve.

During the prediction phase, the central classifier CF's confidence (Δ) and prediction (y) are taken. A hyperparameter threshold (β) is optimally chosen to compare the pairwise difference between the confidence of the CF and every other subsequent neural network of the ensemble ($\Delta^{'} - \Delta$). If the confidence crosses β, the prediction of that neural network is returned.

4.2 Parallel CFDNN

The above model acts as a baseline model for the classification of the breast cancer tumour but CFDNN also realises the scenario where more than one processing unit (node) is available in the system architecture on which it is being executed. We propose two parallel variants (Fig. 2) of the serial CFDNN which leverage themselves based on user requirement. The presence of multiple nodes can either be used to make the system faster or to further optimise the model's performance. In the subsequent subsections, we propose:

1. Parallel CFDNN Version: 1, a learning system that utilises the computing power of multiple nodes to make the system faster with minimum drop in accuracy.
2. Parallel CFDNN Version: 2, a learning system that utilises the computing power of multiple nodes with increase in accuracy with minimum increase in execution/ train time.

Algorithm 1: Serial CFDNN

```
input  : dataset, $depth$, $n$
output: A trained neural network ensemble
Initialization: $gen = dataset, \alpha\ ensemble = \phi, depth = 0$;
Function Serial(gen, α, depth, n)
    if $gen == NULL$ then
        return;
    end
    if $depth > n$ then
        return;
    end
    Create Neural Network = $NN$;
    Train $NN$ with $gen$ using $\alpha$ iterations;
    Adam Optimizer, Softmax Activation;
    Sigmoid Hidden Layer Activation;
    $A_{depth}$ = NULL;
    for $each\ (x, y) \in gen$ do
        if $NN.evaluate\ != y$ then
            $A_{depth} = A_{depth} \cup \{(x, y)\}$;
        end
    end
    ensemble[$depth$] = NN;
    $Serial(A_{depth}, \alpha, depth+1$, n);
end
```

Algorithm 2: Confidence Mapper

```
input  : ensemble, $depth$, $n$, $\beta$
output: Predicted Classification : $y$
Function Mapper(ensemble, n, β, x)
    $\Delta, y$ = ensemble[1].evaluate(x);
    for $i \leftarrow 2$ to $n$ do
        $\Delta^{'}, y^{'}$ = ensemble[$i$].evaluate(x);
        if $\Delta + \beta < \Delta^{'}$ then
            return $y^{'}$;
        end
    end
    return y;
end
```

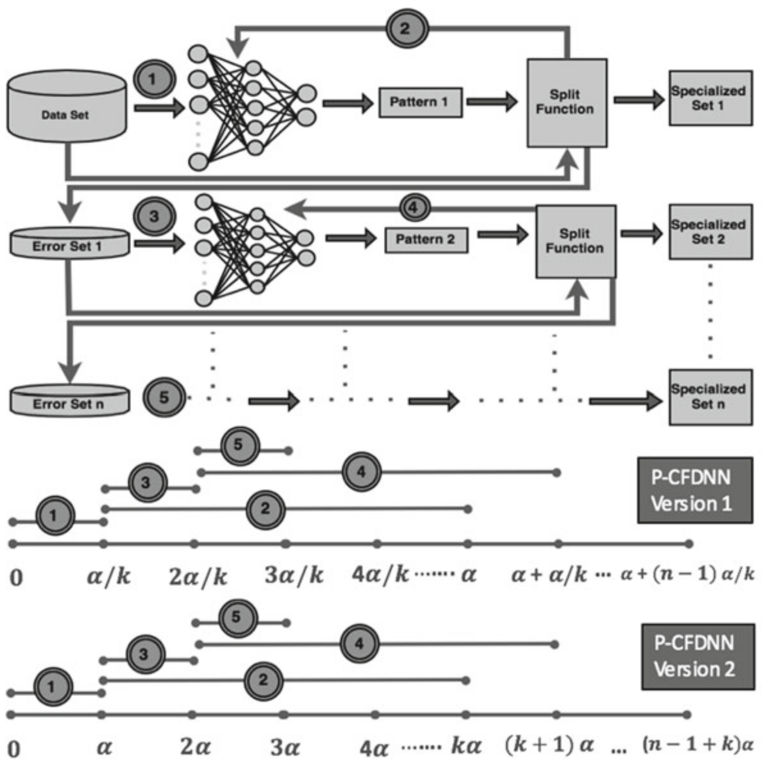

Fig. 2 The *n-depth* parallel CFDNN model. The above timelines represent how the iterations are run simultaneously in the two flavours of the parallel model. Here, 1, 2, 3, 4 and 5 are representative of steps indicating training of the corresponding neural networks with the specified dataset

4.2.1 Parallel CFDNN Version: 1

The central principles of serial CFDNN are adapted with slight modifications to fit the new parallel architecture. Rather than running α iterations before generating the training set of the subsequent neural network, a parent neural network splits the dataset into two sets $D_{\alpha/k}$ and $D'_{\alpha/k}$, where $D_{\alpha/k}$ comprises the training examples that the neural network correctly classifies after just α/k iterations and $D'_{\alpha/k}$ represents the misclassified examples after α/k iterations. Here, k is the scaling factor which controls the number of nodes running in parallel and is set according to computational requirements. The parent neural network then pipelines its remaining iterations ($\alpha - \alpha/k$) with the training of the subsequent neural networks that can now train in parallel with the dataset received earlier ($D'_{\alpha/k}$). This early splitting of the data introduces the following changes in the way CFDNN behaves:

1. At any depth of the ensemble, the size of the misclassified instances dataset ($D^{'}_{\alpha/k}$) is larger for the parallel CFDNN variant when compared to the size of misclassified training instances at the same depth in the serial CFDNN.
2. Likewise, the size of the correctly classified dataset is comparatively smaller at any given depth.
3. Different neural networks of the ensemble are trained on different nodes of the device used.

Algorithm 3: Parallel CFDNN Version: 1

```
input : dataset, $depth$, $n$, $k$
output: A trained neural network ensemble
1  Initialization: $gen$ = $dataset$, $\alpha$ $ensemble = \phi$, $depth = 0$;
2  Function PCFDNN1($gen$, $\alpha$, $depth$, $n$, $k$)
3      if $gen == NULL$ then
4          return;
5      end
6      if $depth > n$ then
7          return;
8      end
9      Create Neural Network = $NN$;
10     Train $NN$ with $gen$ using $\alpha/k$ iterations;
11     Adam Optimizer, Softmax Activation;
12     Sigmoid Hidden Layer Activation;
13     $D^{'}_{\alpha}$ = NULL, $D_{\alpha}$ = NULL;
14     for $each$ $(x, y) \in gen$ do
15         if $NN.evaluate$ != $y$ then
16             $D^{'}_{\alpha} = D^{'}_{\alpha} \cup \{(x, y)\}$;
17         else
18             $D_{\alpha} = D_{\alpha} \cup \{(x, y)\}$;
19         end
20     end
21     Machines[$depth + 1$].execute($PCFDNN1(D^{'}_{\alpha}$, $\alpha$,$depth$+1, n, k));
22     Goto (10): Train with $\alpha - \alpha/k$ iterations with $gen = D_{\alpha}$;
23     ensemble[$depth$] = NN;
24 end
```

These two changes are intuitive results obtained due to the fact that it is a partially learnt neural network that is splitting the data. Since $D^{'}_{\alpha/k}$ can now be a dataset of considerable size, training on it repeatedly may now lead to a loss of accuracy of the neural network over the partial dataset $D_{\alpha/k}$.

This goes against the basic intuition of CFDNN where each neural network should focus to be highly specialised on its assigned dataset $D_{\alpha/k}$ and not to improve its classification accuracy on the entire dataset. Therefore, another introduced difference in the Parallel CFDNN Version: 1 is that the parent network trains only on $D_{\alpha/k}$ after the data split point (i.e. α/k iterations), whereas, in the serial CDFNN, the parent network would train completely on $D_{\alpha/k} \cup D'_{\alpha/k}$, for the complete α iterations. This strengthens its confidence over $D_{\alpha/k}$ and allows us to use the same low-computation-based mapper function that we used in serial CFDNN.

4.2.2 Parallel CFDNN Version: 2

The second parallel variant of CFDNN aims to leverage the increased computation power to increase the performance of the learning system without an exponential increase in execution/training time. The variant 2 increases the number of iteration per neural network to $k\alpha$, instead of α/k like in the variant 1. Here again, k is the scaling factor, but rather than scaling down the number of iterations, it scales up the number of iterations. The split point like serial CFDNN is created after α iterations. The subsequent $(k-1)\alpha$ iterations are pipelined along with the subsequent neural networks like in parallel CFDNN variant 1. The effects of this scaling are as follows. For the breast cancer detection classification problem, $k = 2$ was found to be the optimal scale-up factor.

Algorithm 4: Parallel CFDNN Version: 2

```
   input  : dataset, $depth$, $n$, $k$
   output : A trained neural network ensemble
 1 Initialization: $gen$ = $dataset$, $\alpha$ $ensemble$ = $\phi$, $depth = 0$;
 2 Function PCFDNN2($gen$, $\alpha$, $depth$, $n$, $k$)
 3     if $gen$ == NULL then
 4         return;
 5     end
 6     if $depth > n$ then
 7         return;
 8     end
 9     Create Neural Network = $NN$;
10     Train $NN$ with $gen$ using $\alpha$ iterations;
11     Adam Optimizer, Softmax Activation;
12     Sigmoid Hidden Layer Activation;
13     $D'_\alpha$ = NULL, $D_\alpha$ = NULL;
14     for each $(x, y) \in gen$ do
15         if $NN.evaluate$ != $y$ then
16             $D'_\alpha = D'_\alpha \cup \{(x, y)\}$;
17         else
18             $D_\alpha = D_\alpha \cup \{(x, y)\}$;
19         end
20     end
21     Machines[$depth + 1$].execute($PCFDNN2(D'_\alpha$, $\alpha$, $depth$+1, n, k));
22     Goto (10): Train with $(k-1) \times \alpha$ iterations with $gen = D_\alpha$;
23     ensemble[$depth$] = NN;
24 end
```

Table 1 The comparison of the various flavours of CFDNN with state-of-the-art learning algorithms on WBCD Database (Data Distribution: 70–30)

Algorithm	Average Accuracy	Algorithm	Average Accuracy
TF-GNNE [16]	99.90%	Devi et. al [6]	99.60%
AMMLP [7]	99.26%	**S-CFDNN**	**99.01%**
DBN-NN [8]	99.59%	**P-CFDNN Version: 1**	**98.50%**
RS - BPNN [9]	98.60%	**P-CFDNN Version: 2**	**99.01%**

5 Results

The three variants of the CFDNN algorithm were evaluated on the Wisconsin Breast Cancer Dataset. The models were implemented in Python using TensorFlow. We used 70–30% training to test examples ratio to experiment our models. We used $k = 2$ for both the parallel variants of the CFDNN Algorithm. Each member of the neural network ensemble contained five units in the hidden layer.

An average accuracy of 99.01% was obtained over 10 runs of the serial CFDNN model. The parallel CFDNN version 1 model approaches the serial CFDNN with an average accuracy of 98.5%. The parallel CFDNN version 2 model gets the maximum clocked accuracy of 99.5%, but averages exactly as the serial CFDNN, most probably owing to overfitting due to a small dataset size. Table 1 compares CFDNN with various algorithms tested on WBCD database. Each of these algorithms is computationally highly expensive compared to CFDNN.

6 Conclusions

All the three variants of CFDNN seem to be competing with the state-of-the-art algorithms in literature despite being computationally inexpensive. The mapper function proposed to map the multiple unit outputs to a single ensemble output seems to be fitting the intuitive sense on which the learning model is based; however, given that experimentation has shown considerably low error intersection set even when trained with less data, it is a salient task to implement an intelligent learning-based mapper function as well as a part of our future work. Also, since the amount of data decreases with the depth of the ensemble, computationally inexpensive algorithms that train on sparse data efficiently definitely show a promise in increasing the performance of the system.

References

1. Siegel RL, Miller KD, Jemal A (2016) Cancer statistics, 2016. CA: Cancer J Clin 66:7–30
2. Azami H, Escudero J (2015) A comparative study of breast cancer diagnosis based on neural network ensemble via improved training algorithms. In: 37th Annual International conference of the IEEE engineering in medicine and biology society (EMBC). pp 2836–2839
3. Akay MF (2009) Support vector machines combined with feature selection for breast cancer diagnosis. Expert Syst Appl 36:3240–3247
4. Derya E, Ubeyli (2007) Implementing automated diagnostic systems for breast cancer detection. Expert Syst Appl 33:1054–1062
5. Alickovic E, Subasi A (2017) Breast cancer diagnosis using GA feature selection and rotation forest. Neural Comput Appl 28(4):753–763
6. Devi RDH, Devi MI (2016) Outlier detection algorithm combined with decision tree classifier for early diagnosis of breast cancer. Int J Adv Eng, Chicago. Tech/Vol VII/Issue II/April–June 93:98
7. Marcano-Cedeo A, Quintanilla-Domnguez J, Andina D (2011) WBCD breast cancer database classification applying artificial metaplasticity neural network. Expert Syst Appl 38(8):9573–9579
8. Abdel-Zaher AM, Eldeib AM (2016) Breast cancer classification using deep belief networks. Expert Syst Appl 46:139–144
9. Nahato KB, Nehemiah HK (2015) Kannan A (2015) Knowledge Mining from clinical datasets using rough sets and backpropagation. Neural Netw Comp Math Methods Med 460189(460181–460189):460113
10. Sayed GI et al (2016) Breast cancer diagnosis approach based on meta-heuristic optimization algorithm inspired by the bubble-net hunting strategy of whales. In: International conference on genetic and evolutionary computing, Springer International Publishing
11. Bhardwaj A, Tiwari A (2015) Breast cancer diagnosis using genetically optimized neural network model. Expert Syst Appl 42(10):4611–4620
12. Bache K, Lichman M (2013) UCI machine learning repository, 2013 [Online]. http://archive.ics.uci.edu/ml
13. Opitz DW, Shavlik JW (1996) Generating Accurate and Diverse Members of a Neural-Network Ensemble. Adv Neural Inf Process Syst 8:535–541
14. Rivero D, Dorado J, Rabual J, Pazos A (2010) Generation and simplification of artificial neural networks by means of genetic programming. Neurocomputing
15. Koza JR, Rice JP (1991) Genetic generation of both the weights and architecture for a neural network. In: International joint conference on neural networks (IJCNN-91), vol 2. pp 397–404
16. Singh I, Sanwal K, Praveen S (2016) Breast cancer detection using two-fold genetic evolution of neural network ensembles. In: IEEE 2016 International conference on data science and engineering (ICDSE). IEEE