Breast Cancer Histopathology Image Classification with Deep Convolutional Neural Networks

Steve A. Adeshina[1*] Adeyinka P. Adedigba [2] Ahmed A. Adeniyi [3] & Abiodun M. Aibinu [4]
[1,3]Departments of Computer Sciences and Engineering, Nile University of Nigeria, Abuja
E-mail : steve.adeshina@nileuniversity.edu.ng
[2,4]Department of Mechatronics Engineering, Federal University of Technology, Minna

Abstract - This work addresses the problem of intra-class classification of Breast Histopathology images into Eight (8) classes of either Benign or Malignant Cell. Current manual features extraction and classification is fraught with inaccuracies leading to high rate false negatives with attendant mortality. Deep Convolutional Neural Networks (DCNN) have been shown to be effective in classification of Images. We adopted a DCNN architecture combined with Ensemble learning method using TensorFlow Framework with Backpropagation training and ReLU activation function to achieve accurate automated classification of these Images. We achieved inter-class classification accuracy of 91.5% with the BreakHis dataset.

Index Terms—Deep Convolutional Neural Network (DCNN), Ensemble Learning Cell, breast Cancer, Histopathology Image, Tensorflow framework, Deep Learning.

1 Introduction

Breast cancer is the second leading cause of death among women and the leading cause of death for women between the ages 45 and 55 worldwide [1]. There are several methods of detecting breast cancer. These methods include the use of several imaging modalities such Mamograms, Magnetic Resonance, ultrasound and thermography amongst others. Analysis of histopathology images of the breast is considered critical in identifying any form of melanoma including breast cancer [2]. Histopathology images obtained from a routinely stained biopsy (a surgically removed specimen) using a form of immunohistochemistry e.g. hematoxylin and eosin (H&E) staining method. A specially trained medical personnel known as a pathologist examines the specimen and is able to give a diagnosis [3]. However, as assuring as histopathology images are in detecting any form of cancer in the body, it is a highly tedious task, which would require a vast knowledge and experienced pathologist to give an accurate and precise result. The accuracy can further be impaired by factors like decrease in attention due to fatigue amongst many other factors[2]. This could in turn lead to a case of misdiagnosis. Deep learning on the other hand, has been shown to perform well in areas of pattern recognition, which has led to a wide research in application of deep learning in medical images [1]. Deep learning aims at learning features directly from input images while avoiding handcrafted features. One great improvement of using deep learning over other methods of pattern recognition e.g. texture descriptors like Complete Local Binary Pattern (CLBP) is that it does not require the use of textual descriptors.The model is able to learn these discriminating features on its own.

Spanol *et al.* [3] introduced a breast histopathology image dataset called BreakHis annotated by seven pathologist in Brazil. They further used six different textual descriptors and different classifiers for the binary classification of the images into benign and malignant cells. They reported an accuracy ranging from 80% - 85% using different magnification factors of the images available. Spanol *et al.* [4] Proved that CNN achieves better results compared to handcrafted textual descriptor used in [3] by achieving an improvement of 6% in accuracy of the binary classification of Breakhis[3] Breast histopathology images. They acheived this by using a CNN of three convolutional layers and two fully connected layers. Whole slide images of 700 x 460 pixels were divided into patches of (32 x 32), (64 x 64) to train the network, the image patches were combine together at the end of the training for final prediction of the model. Xiao *et al.* [1] used image patches for binary classification of metastatsis of breast cancer into tumor patches and non-tumor patches. They used a combination of two pre-existing architecture (ResNet and VGG16) and a combination of handcrafted feature with CNN based feature extracted for broad representation of features for classification. Rakhlin *et al.* [5], proposed a solution for the problem of insufficient dataset in training a deep neural network for classification. Deep convolutional feature representation approach with data augmentation was used in lieu of the commonly used fine-tuning approach to classify breast histopathology images into benign and malignant cells. Additionally, LightGBM an implementation of gradient boosted trees was used for the supervised classification [7] which is broadly used in machine learning because of their speed, accuracy and robustness against overfitting [8]. An accuracy of 93.8% was achieved.

The closest work to this work in the literature is that of Spanol *et al.* [3], [4]. Whereas Spanol *et al.* classified the histopathology images into 2 classes, we classified the same images into 8 classes. The uniqueness in this work is that with intra class classification, the type of tumour is easily

determined. This leads to early commencement of treatment without further tests. Additionally we have also introduced a unique DCNN structure. We achieved comparable results with the best related work in the literature. However Spanol *et al.* perform a binary classification we achieved a multi-class classification same BreakHis dataset. We are unaware of a better accuracy result for multi-class classification of Breast Histopathology images with DCNN.

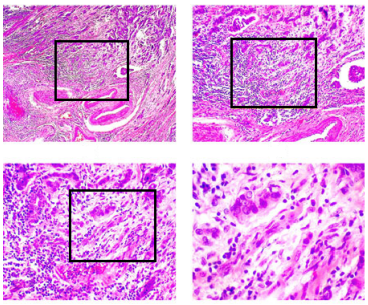

Figure 1: Breast Malignant tumors

Figure 1: slides of Breast Malignant tumors at different magnifications stained with H&E technique. Manually added rectangles to show areas of interest as picked by the Pathologists (ROI identified in lower magnification, zoomed in at different levels for image acquisition) [3].

2 METHODOLOGY

2.1 Dataset

Benign Tumor	
Tumor Type	*Number of images*
Adenosis	111
Fibroadenoma	264
Phyllode Tumor	108
Tubular Adenoma	140
Total	623
Malignant Tumor	
Tumor Type	*Number of Images*
Ductal Carcinoma	896
Lobular Carcinoma	163
Mucinous Carcinoma	196
Papillary Carcinoma	135
Total	1,390
Total Image	2,013

Table 1: Summary of Dataset

2.2 Pre-processing

The images were pre-processed before feeding into the classifier. The images were resized into a 400x400x3 resolution then centralized by normalization. The images were normalized by subtracting the mean from each pixel intensity, this produced an image centered around the mean. The centralized images were then divided by the variance. Mathematically, this is given by:

$I_{mean} = i - \mu \qquad (1)$

$I_{var} = \frac{I_{mean}}{\sigma^2} \qquad (2)$

Where :
I is the unnormalized image
μ is the mean of the image
σ^2 is the variance of the image

Target Encoding: The dataset contains two classes of breast cancer tumour – benign and malignant. Each of these classes contains four subclasses as shown in Table I. For categorical dataset like this, a one hot encoding scheme is used for the target label. Target

$$T_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

Thus, the target label is an 8-dimensional array where each of the dimension represents a class.

2.3 Deep Convolutional Neural Network

A Deep Convolutional Neural Network (DCNN) is used for this research. The model is trained by feeding the normalized input image, which is processed layer by layer in a feed-forward manner to generate a prediction which is compared with the available ground truth target. The error at the output is computed which is the difference between the predicted output and the actual ground truth. The error is then back-propagated through the network. Training process is an iterative process of multiple forward and back-propagation processing until the model minimizes the error to a bearable extent. The summary of the architecture presented in Table II.
In this research, our DCNN model contains three important components: the input, feature extractor and classifier.

- Input: this is the input layer to the model. It defines the dimension of the images to be fed into the network which is 400x400x3 image.
- Feature Extractor: the strength of our proposed model is in the feature extractor used. Unlike earlier proposed methods, features are not hard-coded rather our model generatively extract useful features from input images which are then used for classification. The features are extracted in three different layers of our DCNN, with the low-level features such as line and edges detected and extracted at the layer1 while high level features such as contours and patterns are extracted in layer 3.
- The classifier contains two fully connected layers and a softmax layer. The softmax function is a generalized form of logistic function used for binary classification, it is given as:
  $\sigma(i) = \frac{e^{im}}{\sum_{n=1}^{N} e^{in}}\ for\ m = 1, ..., N(4)$
  The output of our model is the output of the softmax function, which effectively gives the probability of each class given the input. Then the (test) image is assigned to the class with the highest probability.

Table II: Summary of Architecture:

Layer	Type	Filter Size	Stride	Channel	
1st	conv.	3 x 3	2 x 2	32	Feature extractor
	conv.	3 x 3	1 x 1	32	
	conv.+pool	3 x 3	1 x 1	32	
2nd	conv.	5 x 5	1 x 1	64	
	conv.+pool	5 x 5	1 x 1	64	
3rd	conv.+pool	7 x 7	1 x 1	128	
	conv.+pool	7 x 7	1 x 1	256	
FC 1	fully conn.	....	.....	256	Classifier
FC 2	fully conn.	....	.....	8	

Table 2: Summary of Architecture

Table III: Table showing Trainable Parameters

Layer	Filter Size	Channel	No. of Training neuron
1st	3 x 3	32	896
2nd	3 x 3	32	9,248
3rd	3 x 3	32	9,248
4th	5 x 5	64	51,264
5th	5 x 5	64	102,464
6th	7 x 7	128	401,536
7th	7 x 7	256	1,605,888
FC 1	....	256	6,401
FC 2	....	8	1,001
Total			2,187,946

Table 3: Table showing Trainable Parameters

The network trained a total of Two million, one hundred and eight seven thousand, nine hundred and forty six neurons for the intra-class classification task. The architecture of the model is visually represented below

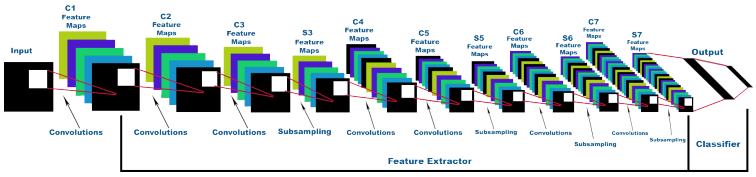

Figure 2: Architecture

2.4 Optimization Algorithms

- Adam optimizer: Adam optimizer is an optimization algorithm for first-order gradient-based optimization of stochastic objective functions, built on adaptive estimates of lower-order moments [9]. It uses the combinative advantages of AdaGrad (which performs well in a situation of spare gradients) and RMSprop (performs well with non-stationary settings). Some of the benefit of Adam optimization are computational efficiency, little memory requirements, invariant to diagonal rescale of the gradients and hyper-parameters possess intuitive interpretation and requires little tunning [9]. The algorithm for the weight update during the back-propagation is given below:

Algorithm 1.

Ada Optimization Algorithm: Our Proposed algorithm for stochastic optization. All operation on vectors are element-wise. with $\beta_1{}^t$ and $\beta_2{}^t$ we denote β_1 and β_2 t the power of t

procedure ADA, $(\alpha : Stepwise)$
Require: β_1 , $\beta_2 \epsilon [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\Theta)$: Stochastic function with parameters Θ
Require: Θ_0 : Initial parameter vector
$m_0 \leftarrow 0$ (Initialize $2^s t$ moment vector)
$u_0 \leftarrow 0$ (Initialize $2^n d$ moment vector)
$t \leftarrow 0$ (Initialize timestep)
while $\Theta_t \neq converge$ **do**
$t \leftarrow t + 1$
$g_t \leftarrow \Delta_\Theta f_t(\Theta_{t-1})$ (Get gradients w.r.t stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$u_t \leftarrow \beta_2 \cdot u_{t-1} + (1 - \beta_2) \cdot {g_t}^2$ (Update biased second raw moment estimate)
$m_t \leftarrow m_t / (1 - {\beta_1}^t)$ (Compute bias-corrected first movement estimate)
$u_t \leftarrow u_t / (1 - {\beta_2}^t)$ (Compute bias-corrected first movement estimate)
$\Theta_t \leftarrow \Theta_{t-1} - \alpha \cdot m_t / (\sqrt{u_t} + \epsilon)$ (Update parameters)
return Θ_t (Resulting parameter)
end

- Ensemble learning method (Adaboost): Ensemble learning methods are meta-algorithms that uses the combination of numerous machine learning algorithms into one predictive model for the purpose of decreasing variance, bias or improve predictions [11]. Ensembles have been proven to attain highly accurate results from the combination of less accurate predictions. Adaboost is a type of ensemble learning algorithm that out performs other ensemble learning method like Bagging, Randomized tress. Given a set of classifiers with set of weights to make a prediction $h(i)$ over a training sample in i iterations, the weighted error of the prediction of each classifier $h(i)$ is computed and used to adjust the set of weights of the corresponding classifier using the base learning algorithm. Adaboost is considered as trying to directly optimize the weighted predictions of the classifiers, thereby making a direct assault on the representational problem. This helps to add more weights to classifiers according to the error rate computed from $h(i)$; there by adding more weights to a classifier with high error rate and little or no weight for classifiers with accurate prediction on the training sample. The final classification task is given by the equation below:

$$h_f(x) \rightarrow \sum_L W_L h_L \qquad (5)$$

Algorithm 2.

Adaboost Algorigthm

procedure ADABOOST($\{w_n\}\ to\ \frac{1}{N}$) ▷ Init data weights
for $m = 1 \rightarrow M$ **do** ▷ We have the answer if r is 0
$y_m(x)\ by\ minimizing\ weighted\ error\ function\ J_m$ ▷ fit a classifier
$J_m = \sum_{n=1}^{N} {w_n}^{(m)} 1\left[y_m(x_n) \neq t_n\right]$
compute $\epsilon_m = \sum_{n=1}^{N} {w_n}^{(m)} 1\left[y_m(x_n) \neq t_n\right] / \sum_{n=1}^{N} {w_n}^{(m)}$
evaluate $\alpha_m = log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$
update the data weights: $w_n^{m+1} = w_n^m \exp\left\{\alpha_m 1\left[y_m(x_n) \neq t_n\right]\right\}$
$r \leftarrow a \bmod b$
return $Y_M(x) = sign\left(\sum_{m=1}^{M} \alpha_m y_m(x)\right)$ ▷ make predictions using final model:
end

- TensorFlow Deep learning Framework
TensorFlow is an open source interface for expressing machine learning algorithms as well as implementing such [10], and the algorithm usable across some given platform with little or no changes. It can be considered a low level API which represents multi-dimensional data array as a Tensor and its computation as a directed graph composed of set of nodes. TensorFlow has a faster programming style, interactive graph based control and so also it is highly optimized for both memory and processor efficient use.

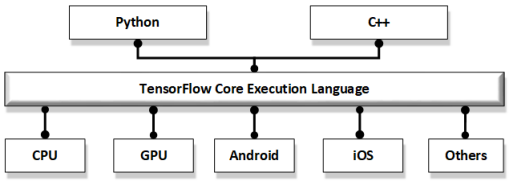

Figure 3: TensorFlow Framework Representation [10].

3 Experiments

3.1 Experimental setup and Parameters

The training of the model was done on a HP Proliant Sever DL360 G7 with a Quad-Core Intel Xeon X5690 3.46 GHz processor, 12 MB Cache memory, 24GB DDR3-1333 RAM and a 1 GB Swap Memory using Tensorflow deep learning framework in Python IDE. The following parameters were used for the model's training.

1) Weight initializer: Glorot Uniform Distribution
2) Learning rate: 0.001
3) First moment decay rate: 0.9
4) Second moment decay rate: 0.999
5) drop out: 0.25
6) Ensemble: 3 homogeneous classifiers

3.2 Training Strategy

The original size of 700 x 460 images were resized to 400 x 400. The training used purely supervised type, which is frequently used in systems for speech and image recognition. Adam optimizer [9] for gradient descent with back-propagation was used to compute the gradients and a batch size of 64 was used to update the network parameters, with a learning rate of 0.001. The model trained for 6,600 iterations, taking approximately 6 days to train. Algorithm for the training process is given below:

1) Load the images and target file into the memory
2) Pick a seed value S for $(0 \geqslant S \geqslant x)$
3) Select images of 64 images at random from all classes and assign them to the present seed value
4) Commence training with images picked at random for epoch x for $1 \geqslant x \leqslant 100$
5) Repeat "line 2 and 3' till (x = 100).

3.3 Testing

Following the training, the model was tested with 600 images; True Positive, True Negative, False positive and false Negative for each of the classes where computed during the testing phase of the experiment which is then used to get the overall Accuracy, precision and recall of the model.

4 Results

In this section, the classification result is presented followed by a brief discussion of the results obtained. The figure below shows the cost function representing the overall error made by the network in the course of training the model. The optimization algorithm is used to minimize the error during the backward pass of the model's training phase which in turn increases the accuracy of the model as the training process continues until the model tends to converge given its highest accuracy at that point. Our model was able to converge at an average cost of 1.85. A feature map at the last fully connected layer was visualized to see what the model was able to learn. The image plotted show neurons with high activation in the middle of the map thus corresponding to the method of acquisition of the Region of Interest (ROI) picked by the pathologist as illustrated in [3] and shown in Figure 1 above with most of the cell tumour differentiating patterns present in the middle of the whole side image (WSI).

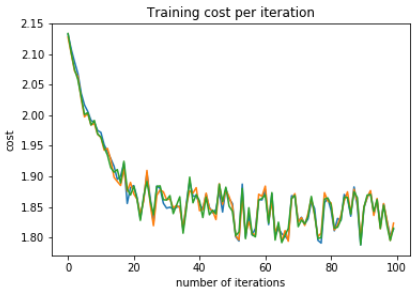

Figure 4: Reduction of Cost per training iteration

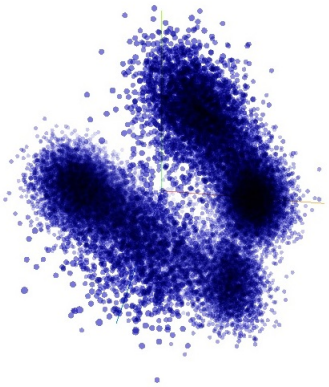

Figure 5: Visual representation of the last fully connected layer learned feature map with high activation in the darker colored areas.

Our result is based on the following conditions for testing Phase for each of the eight classes.

1) True Positive (TP) - detects condition when condition is present.
2) True Negative (TN) - detects no condition when condition is not present.
3) False Positive (FP) - detects condition when condition is not present.
4) False Negative (FN) - detects no condition when condition is present.

Below is the confusion matrix of the model showing the values for True positive, True Negative, False Positive and False Negative for the eight class prediction

Figure 6: Confusion Matrix

The following performance evaluation metrics were used for evaluating the model: accuracy, error rate, precision, recall.
Model Accuracy: The model accuracy gives the measure of correct prediction of the model compared to the overall data point. It is given by:

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (5)$$

Error Rate: gives the measure of wrong class prediction of the model compared to the overall data-point. It is given by:

$$Errorrate = \frac{(FN+FP)}{(TP+TN+FP+FN)} \quad (6)$$

Precision: it is not enough to know the accuracy of the model, the precision of the model gives the proportion of the predicted positive which is actually positive. This is important in cancer detection, because early warning of detection of cancer can increase the possibility of its cure. The model precision is given as:

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

Recall: this gives the proportion of positive which are predicted as positive. It is given by:

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

Therefore, using 5 to 8, Table 4 shows that the model accuracy is 91.54%, error rate is 8.45%, precision is 63.36% and recall rate is 76.67%

Accuracy	Error rate	Precision	Recall
91.54%	8.45%	63.36%	76.67 %

Table 4: Table showing evaluation metrics

5 Discussions And Conclusion

5.1 Conclusion

In this work, we presented a method for the classification of the BreakHis Breast Histopathology dataset into different tumor types with intra-class classification. This was acheived by using a task specific deep learning neural network architecture. An overall accuracy of 91.5% was recorded. There is need for this intra-class classification which will help in determining the particular tumor type. The commonly proposed binary classification will require additional activities and tests before the tumour type is determined.

The results acheived in this work compare favourably with what exist in the literature. The closest work to ours is that of Spanol *et al.* [3], [4]. Whereas Spanol *et al.* achieved an accuracy of 93.5% with binary classification, we acheived a multi-class classification accuracy of 91.6% using the same dataset. We are unaware of a better result for multi-class classification in the literature.

Additionally the use of whole slide images (WSI) for feature extraction will reduce the computation time and resources as compared to the division of the images into patches to train/test and the combination of this images back for final prediction.

In other to enhance the current accuracy and other performance metrics, a much deeper neural network architecture is being considered. Such architecture will have the capability of learning more discriminating features between the classes. Additionally the use of heterogeneous ensemble learning method is being considered. This will allow the use of different classifiers for the ensemble thereby creating more variants between the ensembles, making the model generalize

Acknowledgments

The authors would like to thank everyone who contributed to the success of this work...

References

[1] K. Xiao, Z. Wang, T. Xu and T. Wan *"A Deep Learning Method For Detecting And Classfying Breast Cancer Metastasis In Lymph Nodes on Histopathological Images"*. Beijing, 2017

[2] J. E. Joy, E. E. Penhoet and D. B. Petitti *"Saving Women's Lives: Strategies for Improving Breast Cancer Detection and Diagnosis" National Academies Press, Washington D.C., 2005.*

[3] F. A. Spanhol, L. S. Oliveira, C. Petitjean and L. Heutte *"Breast Cancer Histopathological Image Classification using Convolutional Neural Networks," in International Joint Conference on Neural Networks*. Brazil, 2016.

[4] F. A. Spanhol, L. S. Oliveria, C. Petitjean and L. Heutte *"A Database for Breast Cancer Histopathological Image Classification" IEEE Transactions on Biomedical Engineering, vol. 63, no. 7, pp. 1455-1462, 2016.*

[5] A. Rakhlin, A. Shvets, V. Iglovikov and A. A. Kalinin *"Deep Convolutional Neural Networks for Breast Cancer Histology Image Analysis" Cornell University Library, Computer Science, Computer Vision and Pattern Recognition* ioRxiv, p. 1802.00752, 3 April 2018.

[6] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu and M. S. Lew *"Deep learning for Visual understanding: A Review" Neurocomputing 187, pp. 27-48, 2016.*

[7] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T. Y. Liu *"A highly efficient gradient boosting decision tree" Advances in Neural Information Processing Systems* pp. 3149-3157, 2017.

[8] A. Natekin and A. Knoll *"Gradient Boosting Machines; a tutorial" Frontiers in Neurorobotics .*

[9] D. P. Kingma and J. Ba *"Adam: A method for stochastic optimization" in ICLR, 2015.*

[10] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu and Xiaoqiang Zheng, *"TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems" Google Research; White Paper, 2015.*

[11] J. Thongkam, X. Guandong and Z. Yanchun *""AdaBoost Algorithm with Random Forests for Predicting Breast Cancer Survivability,". in Proceedings of the International Joint Conference on Neural Networks retrieved at Feb 22nd 2017*