Improved Transfer Learning through Shallow Network Embedding for Classification of Leukemia Cells

Kaushik S Kalmady[1], Adithya S Kamath[1], Gopakumar G[2], Gorthi. R. K. Sai Subrahmanyam[3], Sai Siva Gorthi[4]

Abstract— **One of the most crucial parts in the diagnosis of a wide variety of ailments is cytopathological testing. This process is often laborious, time consuming and requires skill. These constraints have led to interests in automating the process. Several deep learning based methods have been proposed in this domain to enable machines to gain human expertise. In this paper, we investigate the effectiveness of transfer learning using fine-tuned features from modified deep neural architectures and certain ensemble learning methods for classifying the leukemia cell lines HL60, MOLT, and K562. Microfluidics-based imaging flow cytometry (mIFC) is used for obtaining the images instead of image cytometry. This is because mIFC guarantees significantly higher throughput and is easy to set up with minimal expenses. We find that the use of fine-tuned features from a modified deep neural network for transfer learning provides a substantial improvement in performance compared to earlier works. We also identify that without any fine tuning, feature selection using ensemble methods on the deep features also provide comparable performance on the considered Leukemia cell classification problem. These results show that automated methods can in fact be a valuable guide in cytopathological testing especially in resource limited settings.**

I. Introduction

Cytopathology [1] refers to a diagnostic technique that uses cells as a base for disease detection. It is a very critical step in chronic disease diagnosis like cancer. A very conventional way of carrying out the test consists of smearing of samples across a glass microscope slide and examining them under a powerful microscope when stained. Though this method of cellular level investigation assures detailed images with special localization of sub-cellular components, it requires a lot of manual labor and is associated with low throughput. Efforts have been made to automate the process of slide preparation and subsequent analysis through the use of robotics. But robotics, being an expensive mode of automation, raises the cost of the system significantly, if used for slide imaging-based diagnosis. Microfluidic Imaging Flow Cytometry (mIFC) [2] is an emerging method of microscopic imaging. It aims to reduce the complexity of the tasks involved in cytometry by combining flow cytometry with digital microscopy thus combining merits (statistical power and quantitative morphology) of both techniques. It involves making cells in suspension and allowing the flow of cells through microfluidics channel where high resolution video frames are imaged and morphological features are captured. These steps only take few minutes as opposed to the steps in manual diagnosis that take hours of work. Hence, mIFC has gained importance owing to its high-throughput, low cost and effectiveness in settings where the resources are limited.

After the images are captured, the next step would be to build an automated system that can simulate the skill of a trained cytopathologist. There has been considerable work in building systems that aid the cytopathologist in making more informed decisions. Most of the systems use size, shape and structure of the cells as criteria to obtain class labels for cancerous cells [3]. Several Neural Network (NN) based methods have also been deployed [4][5]. In recent years, deep learning based methods have proven to provide excellent results in the domain of image classification without there being any need for hand engineered features. In particular transfer learning methods have been found to be of excellent aid in medical image processing for melanoma screening [6] and lung pattern analysis [7].

Earlier works [9] have investigated the effectiveness of latent features extracted from deep neural networks pre-trained on the popular imaging database, ImageNet [8] for the classification on leukemia cell images captured using mIFC. These features are highly discriminative and provide superior performance to existing methods. In this paper, we further improve on this performance by employing fine-tuning to inculcate finer domain knowledge in the deep neural network. We further explore some ensemble learning techniques for classification using latent features with a number of weak classifiers. The following are the major contributions of this work: (i) We demonstrate that using a shallow Feed Forward Neural Network (FFNN) instead of an SVM for classification using latent features not only provides comparable or better results but also provides us with knowledge rich neural network layers that can be directly embedded into deep neural networks like VGG16 for further fine tuning (ii) We find that when the trained FFNN layers are embedded into VGG16 and the hierarchically deeper layers of VGG16 are fine-tuned, an

[1]Department of Computer Science, NITK, Surathkal
[2]Department of Computer Science, Amrita Vishwa Vidyapeetham, Amritapuri
[3]Department of Electrical Engineering, IIT Tirupati
[4]Department of Instrumentation and Applied Physics, IISc, Bangalore

978-1-5386-2241-4/17/$31.00 ©2017 IEEE

improvement in accuracy is obtained even with very less training data (iii) Without considering the fine-tuning of the deep features we have explored the effectiveness of ensemble techniques like Bagging and Stacking on VGG16 and ResNet50 features and demonstrate that they provide performance comparable to that of fine-tuning. To further demonstrate the effectiveness of fine-tuned features over directly extracted features, we investigate SVM classification on fine-tuned features and compare the performance with previous works. Specifically we experiment with the three important classes of leukemia cell-line images: HL60, MOLT, and K562. Our results and analysis show that the FFNN classification gives 99.52% accuracy. Embedding the FFNN layers into VGG16 and fine-tuning the only the hierarchically deeper layers provides considerable improvement in performance over the direct use of features even with minimal training. With 5% training data, accuracy has been observed to improve from 91.43% [9] to 98.96%. Ensemble methods yield comparable performance even without need of fine tuning, however requiring more testing time.

II. IMAGING SCHEME FOR THE DATASET

A challenging part in this mIFC based diagnosis is the segmentation of cells. mIFC, being a recently ventured topic, has not seen well-established algorithms in case of image segmentation and localization mainly because of computational complexity. Rather than dealing with finely segmented cell images, we use roughly segmented cells. Fig.1 shows K562, MOLT and HL60 cell lines flowing through the microfluidic channels. Fig.2. shows each of the roughly localized cells identified by a bounding box. To obtain quality images whose features are sufficient to distinguish between images of different classes, de-noising is done with the help of a 5×5 average mask. The image quality is further improved by background subtraction, where the background image frame is obtained by capturing only the sheath fluid flow through the channel. The detailed procedure can be found in [9].

The above step is followed by discarding of frames that either do not contain cells or contain objects of size less than that of cell. Rough segmentation of the obtained images is carried out by using regional properties and morphological features. The process is explained in detail in [13]. After these steps, a binary image of cell in the frame is obtained and a rough bounding box is used to enclose every cell and the resulting images are used as the dataset for further experiments.

III. CNNs FOR FEATURE EXTRACTION

Convolutional Neural Networks (CNN) have been found to be extremely effective for the task of image classification in a wide range of fields. Further, their ability to recognize localized features has been utilized in medical domain for interstitial lung disease detection [14], volumetric MRI [15], and radiotherapy [16]. The transfer learning capabilities of CNNs have been studied and used in a range of applications from malaria analysis [17] to ultrasound mammography [18].

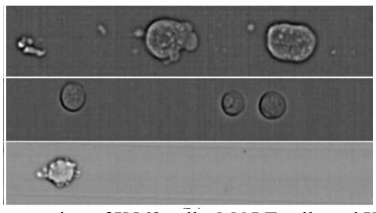

Fig 1: Rough segmentation of K562 cells, MOLT cells and HL60 cells

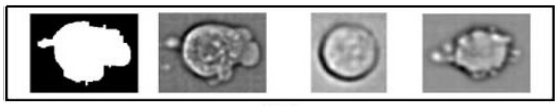

Fig 2: Fig 2: Localization with bounding box on cells in Fig. 1.

A typical CNN is made of the following basic components, (a) set of convolutional layers, (b) ReLU (Rectified Linear Unit) activation function, and (c) Max-pooling layer. These components are laid out one after the other, as described in [19]. For most of our experiments we use features extracted from few final layers of VGG16. We also investigate features extracted from the last pooling layer of ResNet50.

VGG16 is a popular CNN architecture originally designed to classify images in the ImageNet dataset. It's design can be thought of as made up multiple blocks containing 3×3 convolutions and 2×2 max pooling layers. Final output from block 5 is fed to a fully connected classifier at the end. The entire network can be divided into 5 blocks with block 1 and 2 having 2 convolution layers and a max-pooling layer, and block 3, 4, 5 having 3 convolution layers and max-pooling layer. The fully connected classifier has two fully connected layers (fc_1 and fc_2) containing 4096 neurons and finally a softmax layer for 1000 classes. Outputs from every convolution layer are fed as input to a ReLU Activation function.

ResNet50 is another popular CNN Architecture which consists of repeated block of 3×3 convolutions backed by identity mapping. This block is referred to as residual layer, which is claimed to allow easy backward flow of gradients directly through identity mapping during backpropagation. The down-sampling effect of max-pooling layers is brought with the use of 3×3 convolutions of a different stride when compared to the previous convolution layers. Here, stride is the number of steps skipped by any convolution while passing over the feature map. Only two pooling layers are used, one during initial stages and the other, called 'Global Average Pooling' (GAP), at the end (before softmax classifier), which is a substitute to the traditional fully connected layers used in other CNN models. More details on ResNet50 architecture can be found in [20]. For our experiments, we extract features from GAP layer (avg_pool) of ResNet50 pre-trained on ImageNet and run classifiers on the features.

IV. METHODS AND OBJECTIVES

A. FFNN Training on VGG16 features

Outputs from the intermediate layers of VGG16 are used to train shallow FFNNs. For our purposes, we use the outputs of fc_1, fc_2 and the final convolution and pool layers in block 5 ($conv_5$_$block_3$ and $conv_5$_pool) of VGG16 architecture. The

objective is to not only demonstrate the discriminative nature of these features but also have layers that hold valuable knowledge about our dataset. These layers are then embedded into VGG16 for faster fine tuning. We initialize VGG16 layers to weights trained on ImageNet, which is a set of several hundreds of non-medical images belonging to over 1000 class (like cars, birds etc.). For each image in our dataset, we feed it to the network and take outputs from fc_1, fc_2, $block_5\_conv_3$, $block_5\_pool$ as feature representations and separately train a FFNN on each of them. Outputs from fc_1 and fc_2 are 4096 dimensional vectors but $block_5\_conv_3$ produces 512 feature maps of shape 14×14 and $block_5\_pool$ produces 512 feature maps of shape 7×7. These feature maps are flattened before being passed as input to the FFNN which has two hidden layers containing 1024 and 256 neurons, which were chosen after experimenting with incremental values of 64, 128, 256 until learning did not improve, and finally a softmax layer with 3 neurons. The numbers of input neurons are changed according to the dimensionality of the features being trained on. A FFNN with identical parameters for each of the features is trained and performance is noted. The results show that these latent features enable the model to converge to very high accuracies very fast as compared to raw image input. Also, we experiment by taking the features that give the best results and run further tests with lesser amount of training data to examine speed of convergence and model accuracy, all on the same configuration of FFNN under similar experimental parameters.

B. *Fine Tuning layers of modified VGG16*

We embed the trained FFNN into VGG16 by attaching it to the final layers of VGG16 and fine-tune the hierarchically deep layers, namely the $block_5\_conv_3$ and $block_5\_pool$ layers on our dataset of microscopic images. First, VGG16 layers are initialized to ImageNet weights. We then do two modifications to VGG16, (i) We remove the fully connected classifier (fc_1, fc_2, final softmax layer) at the end of VGG16 and replace it with the FFNN trained on $block_5\_pool$ layers (which is observed to give the best results). Since the layers in the FFNN have been trained effectively represent our data, we expect them to help speed up convergence and learning. (ii) We allow training only for the final $block_5\_conv_3$ and $block_5\_pool$ layers. The intuition is that since the FFNN already contains information about the data in its hidden layers, it will improve the rate of convergence of the deep neural network and help its final layers learn finer details about the domain specific features with less data and little training. During backpropagation, the updating of weights will take place only until these two layers, the weights of all the prior layers are left untouched. Since we already have very high accuracy as seen in Table I, we are only interested in helping the final layers of the model learn a bit more about our data. This also helps in faster training since gradients don't have to be backpropagated till the first hidden layer.

C. *Effectiveness of fine-tuned layers*

The effects of fine tuning is demonstrated by using the features from the block5_pool layer of our fine-tuned VGG16 model for SVM classification and comparing the accuracy with results obtained in [9]. The improvements in the results demonstrate that the fine-tuning has in fact helped these hidden layers learn better discriminative representations specific to our data. We also train FFNNs as in Section IV A. but this time with fine-tuned features as input, providing very less data for training and observe performance.

D. *Ensemble Learning*

These are the techniques used to combine the merits of different base classifiers to obtain a better meta-classifier. Many techniques including model-averaging and voting are used to combine the models. We extract features from $block_5\_pool$ layer of our VGG16 model (before fine-tuning) and also avg_pool layer of ResNet50 model and run ensemble algorithms like Boosting [10], Bootstrap aggregating (Bagging) [11], Stacking and Voting.

Boosting uses a set of classifiers that do not work well on a given training set and combines then to form a classifier that gives better results on the same training set. They work mainly on reducing bias of the model. Most of the boosting algorithms work in a serial additive way. That is, the model at any point in the algorithm gives more emphasis to the data points that were wrongly classified by the previous model in the sequence of models. This is done by using sample weights on input to each model. The final classification is obtained as a weighted average of the individual models, where the weight for each model is learnt while reducing a loss function defined by the algorithm. The sample weights on input are also obtained during the process of reducing loss function. Two of the most well-established boosting algorithms are AdaBoost and Gradient Boosting Machine (GBM). AdaBoost technique uses exponential loss as its loss function. Whereas GBM technique can use any arbitrary differentiable loss function and uses gradient descent to reduce the loss function. GBM can be thought of as a generalized version of AdaBoost.

Bagging refers to usage of a different subset of input samples each time to train a model, so that the variance of the model is reduced. The final prediction may be obtained by averaging the predictions of models or by voting (majority-rule).

Stacking involves usage of a meta-classifier that learns a combination of predictions of individual base classifiers to give best possible result. Although any of the above mentioned ensemble algorithms can be seen as stacking with an arbitrary meta-classifier, it is a common practice to use simple base classifiers (like SVM, KNN, Logistic Regression, etc.) as a meta-classifier.

Soft voting refers to a simple majority-rule based voting technique done among different base models. It can either work directly on predictions of each model or work on the class-wise probabilities given by each model.

V. Results and Discussions

We discuss the result obtained for each of the experiment detailed in section III. As noted in section I, the original dataset used are the roughly segmented leukemia cell-line images, which consists of altogether 618 Leukemia cells (388 HL60, 106 MOLT, 124 K562). In order to have a larger dataset, we augment the original dataset with horizontal, vertical flips and rotations (90, 180, and 270 degrees). In

addition to this, we make sure that training and test splits contain the same percentage distribution of images from each class to avoid the model for being skewed towards HL60 which has relatively larger number of images and make sure that images from all classes are used during training. Altogether, 3708 cells (2328 HL60, 636 MOLT, 744 K562,) were used after augmentation. The effectiveness of classifying these cells using features from various layers on different models (FFN, SVM and Ensemble Learning) is demonstrated using accuracy of predictions on the test and training set. We compare the obtained classification accuracy with that of the system described in [9]. All the simulations were performed on a NVIDIA GeForce GTX 780 4GB GPU with Anaconda Python3.

A. FFNN training on VGG16 features

Training was done for 50 epochs using RMSprop optimizer and cross entropy loss, learning rate of 0.0001, using a batch size of 50. 80% of the data from each class was used for training and 20% for testing. The entire experiment is repeated 10 times for each of the different input and the average mean and standard deviation is calculated. The results are summarized in Table I. We also present the results for the raw image data under the same experimental conditions for comparison in Fig. 3 and Fig.4. We observe that features enable the model to converge to high accuracies within 10 epochs with steady decrease in loss as compared to raw image as seen in Fig. 4.

B. Fine Tuning Modified VGG16

After applying modifications as mentioned earlier, VGG16 now has the FFNN trained on $block_5$_pool layer instead of its fully connected classifier at the end, and we make sure that training takes place only until the $block_5$_$conv_3$ and $block_5$_pool layers. From each class we use 80% images for fine-tuning and the remaining for testing. Each input is resized to a fixed shape of 224×224 and provided to the modified VGG16 network for training. We note down the training and testing accuracy after 50 epochs of training and observe that the model begins to converge towards high accuracies from the very beginning as seen in Fig. 5. This can be owed in part to the pre-trained FFN which has conserved representation of our data and helps the modified VGG16 to converge to high accuracies within the first epoch. Testing accuracy increases up to **99.86%** in this case.

C. Effectiveness of fine-tuned feature

We now examine the effectiveness of fine-tuning by extracting features from the fine-tuned block5_pool layers.

TABLE I
FFNN TRAINING AND TESTING ACCURACIES WITH DIFFERENT FEATURES

Input	Training Accuracy	Testing Accuracy
Raw Image Data	62.76 (0.001)	62.83(0.001)
fc_1 features	99.96 (0.0004)	99.17(0.002)
fc_2 features	99.98 (0.00015)	99.04(0.0015)
$block_5$_$conv_3$ features	99.91 (0.0007)	99.37(0.002)
$block_5$_pool features	**99.99 (0.0001)**	**99.52(0.0017)**

We demonstrate that the features from the fine-tuned layers are very effective in SVM classification. The tests were repeated 10 times with k-fold cross validation and average accuracy and standard deviation were noted down. We observe an enhancement in classification accuracy in comparison with [9]. The results are summarized in Table II.

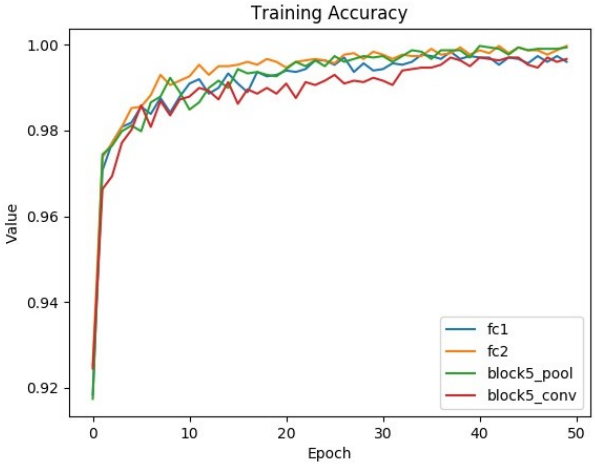

Fig 3: Training Accuracy for 50 epochs of FFNN training

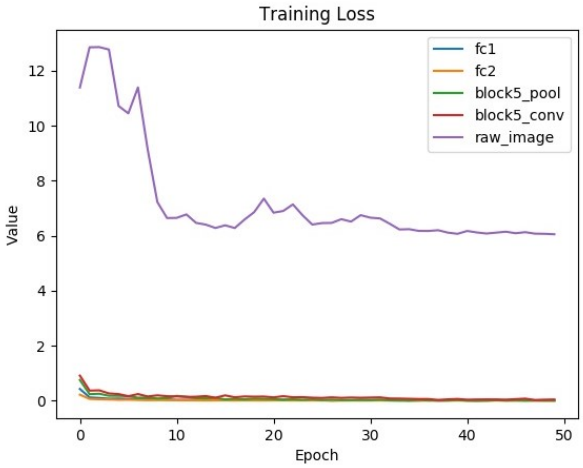

Fig 4: Training loss for 50 epochs of FFNN training

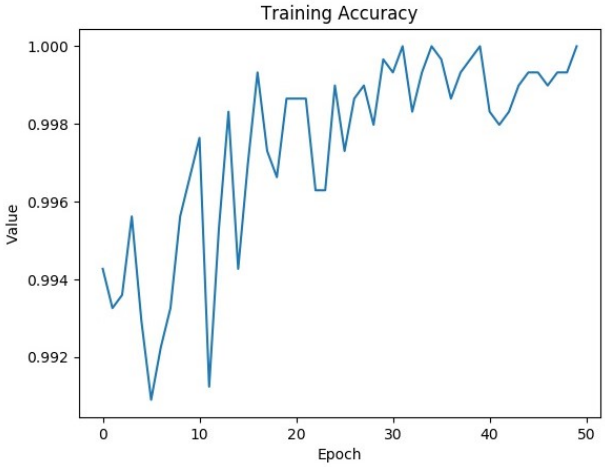

Fig 5: Training accuracy for 50 epochs of VGG16 fine-tuning

We further demonstrate that these fine-tuned layers are able to provide very high accuracy even with small amounts of training data. We train the earlier FFNN with fine-tuned features with 5%, 10%, 20%, 40% training data and compare results with those obtained in [9] in Table III.

D. Ensemble Learning

We use $block_5$_pool features of VGG16 before fine-tuning and avg_pool layer features from ResNet50 as input to test various ensemble methods for cross-validation accuracy and execution time against base classifiers. The following gives a brief description of the ensemble architecture.

At level-0 (base classifiers), three classifiers are used – SVM (with linear kernel), KNNs (with Minkowski distance) and decision trees (of depth 2).

We use 1 SVM, 10 KNNs and 100 decision trees. At level-1(first level of meta-classifiers), two ensemble classifiers are used – bagging and GBM. Bagging runs on KNNs (10 in number) as base classifiers and for each of those KNNs, 60% of the whole dataset is given as input with replacement. GBM runs on decision trees (100 in number) as base classifiers and uses deviance as the loss function. At level-2 (second level of meta-classifiers) single ensemble classifier is used, which is soft voting. Soft voting is run on SVM (from level-0), bagging (from level-1) and GBM (from level-2) for final classification. Soft voting is done using majority-rule based voting in two ways. One is using the class- wise probabilities and the other is using the class predictions, both given by its base classifiers.

We run k fold validation test with k = 2, 3, 4, 5, 8, 10 on the classifiers above mentioned ensemble architecture and obtain an improved accuracy of 99.22% using voting at the cost of execution time. These results are summarized in Table IV and Table V.

TABLE II
SVM CROSS-VALIDATION ACCURACY

Kfold	**10**	**5**	**4**	**3**	**2**
Features in [9]	97.60 (0.48)	97.11 (0.62)	96.50 (0.60)	95.52 (0.20)	94.51 (0.17)
Fine-tuned VGG	**99.89 (0.13)**	**99.84 (0.13)**	**99.70 (0.12)**	**99.73 (0.10)**	**99.60 (0.08)**

TABLE III
FFNN TESTING ACCURACIES WITH LESS DATA FOR TRAINING

Training Data	**CNN Features [9]**	**$block_5$_pool features from fine-tuned VGG16**
5%	91.43(3.33)	98.96(0.24)
10%	95.00(1.40)	99.09(0.22)
20%	96.36(0.97)	99.40(0.17)
30%	96.77(0.76)	99.57(0.16)
40%	96.93(0.90)	99.63(0.12)

TABLE IV
CROSS-VALIDATION ACCURACY – ENSEMBLE LEARNING- VGG16 FEATURES

k-fold	**K=10**	**K=5**	**K=4**	**K=3**	**K=2**
GBM	99.08	98.98	98.92	98.65	98.81
SVM	98.84	98.70	98.70	98.76	98.57
Bagging	98.98	98.89	98.84	98.87	98.49
Voting with probabilities	99.19	**99.11**	**99.11**	**99.22**	**99.16**
Voting with predictions	**99.22**	**99.11**	99.03	99.08	98.92

TABLE V
CROSS-VALIDATION ACCURACY-ENSEMBLE LEARNING - RESNET50 FEATURES

K-fold	**K=10**	**K=5**	**K=4**	**K=3**	**K=2**
GBM	99.19	99.16	99.22	99.22	98.92
SVM	**99.78**	**99.81**	**99.78**	99.60	**99.73**
Bagging	99.46	99.46	99.46	99.46	99.43
Voting using probabilities	99.76	99.76	**99.78**	99.70	99.68
Voting using predictions	99.68	99.78	99.76	**99.73**	99.70

VI. CONCLUSION

A transfer learning and ensemble learning based approach for the automation of cytopathological analysis of Leukemia cell-line images HL60, MOLT, and K562 is proposed. We first train a shallow FFNN with just 2 hidden layers for classification using direct features as in [9]. We then embed this FFNN in VGG16 and fine-tune the hierarchically deep layers. Using features from fine-tuned layers of a modified VGG16 network substantially improves the classification performance compared to direct use of features as in [9]. Also, we explore the use of ensemble learning based feature selection from deep features, which also accomplish comparable performance. Further, we note that since the layers of the modified network have been fine-tuned on a dataset of microscopic images they may be used as efficient feature extractors when compared to VGG16's inherent layers for future applications in microscopic image analysis. We hope that the work done improves the efficiency of cytopathological analysis in resource constrained environments.

VII. ACKNOWLEDGEMENTS

Dr. Sai Siva Gorthi gratefully acknowledges the support extended, during his postdoctoral research work, by Dr. Ethan Schonbrun and Dr. Dian Schaak, of Rowland Institute at Harvard. It helped him in acquiring the data sets of different cell lines used in this work.

REFERENCES

[1] R. Nayar, "Cytopathology in Oncology," *Springer*, 2014.

[2] E. Schonbrun, S. S. Gorthi, and D. Schaak, "Microfabricated multiple field of view imaging flow cytometry," *Lab. Chip*, 12, 2012, pp. 268–273.

[3] J. P. Thiran and B. Macq, "Morphological feature extraction for the classification of digital images of cancerous tissues," *IEEE Trans. Biomed. Eng.* vol. 43, 1996, pp. 1011–1020.

[4] Z. Shi and L. He, "Current status and future potential of neural networks used for medical image processing," *J. Multimedia*, 6, 2011, pp. 244–251.

[5] A. Pouliakis, E. Karakitsou, N. Margari, P. Bountris, M. Haritou, J. Panayiotides, D. Koutsouris, and P. Karakitsos, "Artificial neural networks as decision support tools in cytopathology: past, present, and future," *Biomed. Eng. Comput. Biol.* 7, 2016, pp. 1–18.

[6] A. Menegola, M. Fornaciali, R. Pires, F. V. Bittencourt, S. Avila and E. Valle, "Knowledge transfer for melanoma screening with deep learning," *IEEE 14th International Symposium on Biomed. Imaging (ISBI 2017)*, pp. 297–300.

[7] S. Christodoulidis, M. Anthimopoulos, L. Ebner, A. Christe and S. Mougiakakou, "Multi-source transfer learning with convolutional neural networks for lung pattern analysis," *IEEE Journal of Biomed. and Health Informatics*, vol.21, 2017, pp. 76–84.

[8] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li and F. F. Li, "ImageNet: a large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[9] G. Gopakumar, K. H. Babu, D. Mishra, S. S. Gorthi and G. R. K. S. Subrahmanyam, "Cytopathological image analysis using deep-learning networks in microfluidic microscopy," *Journal of the Optical Society of America A*, vol. 34, 2017, pp. 111–121.

[10] Y. Freund and R. E. Schapire, "A Short Introduction to Boosting," *Journal of Japanese Society for Artificial Intelligence*, 14, 1999, pp. 771–780.

[11] L. Breiman, "Bagging Predictors," *Machine Learning*, 24, 1996, pp. 123–140.

[12] D. H. Wolpert, "Stacked Generalization," *Neural Networks*, 5, 1992, pp. 241–245.

[13] G. Gopakumar, V. K. Jagannadh, S. S. Gorthi and G. R. K. S. Subrahmanyam, "Framework for morphometric classification of cells in imaging flow cytometry," *Journal of Microscopy*, vol. 261, 2016, pp. 307–319.

[14] Q. Li, W. Cai, X. Wang, Y. Zhou, D. Feng, and M. Chen, "Medical image classification with convolutional neural network," *13th International Conference on Control Automation Robotics Vision* (ICARCV 2014), pp. 844–848.

[15] F. Milletari, N. Navab, and S. A. Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," *Fourth International Conference on 3D Vision*, 2016.

[16] N. Zhu, M. Najafi, S. Hancock, and D. Hris, "Deep Convolutional Neural Network Image Matching for Ultrasound Guidance in Radiotherapy," *Medical Physics,* vol. 43*,* 2016, pp. 3301.

[17] Z. Liang, A. Powell, I.Ersoy, M. Poostchi, K. Silamut, K. Palaniappan, P. Guo, Md A. Hossain, A. Sameer, R. J. Maude, J X. Huang, S. Jaeger, G. Thoma, "CNN-based image analysis for malaria diagnosis," *IEEE International Conference on Bioinformatics and Biomedicine(BIBM 2016)*, pp. 493–496.

[18] R. K. Samala, H. P. Chan, L. Hadjiiski, M. A. Helvie, J. Wei, and K. Cha, "Mass detection in digital breast tomosynthesis: Deep convolutional neural network with transfer learning from mammography," *Medical Physics,* vol. 43, 2016

[19] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[20] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.