White Blood Cell Classification Using Convolutional Neural Network

Mayank Sharma, Aishwarya Bhave and Rekh Ram Janghel

Contents

Abstract The density of white blood cells in bloodstream provides a glimpse into the state of the immune system and any potential risks such as heart disease or infection. A dramatic change in the white blood cell count relative to your baseline is generally a sign that your body is currently being affected by an antigen. A variation in a specific type of white blood cell generally correlates with a specific type of antigen. Currently, a manual approach is followed for white blood cell classification; however, some semi-automated approaches have been proposed which involves manual feature extraction and selection and an automated classification using microscopic blood smear images. In this work, we propose deep learning methodology to automate the entire process using convolutional neural networks for a binary class with an accuracy of 96% as well as multiclass classification with an accuracy of 87%.

M. Sharma (✉) · A. Bhave · R. R. Janghel
National Institute of Technology, Raipur, India
e-mail: mayanksharma.nitrr@gmail.com

A. Bhave
e-mail: aishwaryabhave54@gmail.com

R. R. Janghel
e-mail: rrjanghel.it@nitrr.ac.in

J. Wang et al. (eds.), *Soft Computing and Signal Processing*, Advances in Intelligent Systems and Computing 900, https://doi.org/10.1007/978-981-13-3600-3_13

Keywords Leukocytes · White blood cell classification · Convolutional neural network · Deep learning · Transfer learning

1 Introduction to White Blood Cell Classification

White blood cell classification [1–3] deals with two objectives: first, given a stained image of a white blood cell, classify it as either polynuclear or mononuclear. Note that Eosinophils and Neutrophils are polynuclear while Lymphocytes and Monocytes are mononuclear [4]. Second, given a stained image of a white blood cell, classify it as Eosinophils and Neutrophils, Lymphocytes and Monocytes. White blood cells can be distinguished from other cells due to the presence of nucleus . WBCs can be further identified into different types from their nuclear structure.

2 Literature Review

Machine learning algorithms such as k-nearest neighbours, learning vector quantization [5] and support vector machines [6, 7] have been applied over extracted features of image. Extensive image processing algorithms [2, 8] have been applied to extract these features as well as for feature selection. Recently, ANN [1] has been applied over the extracted features of 70 white blood cell images. In this work, we try to automate this feature extraction [9] and selection process in neural networks by using convolutional neural network and also an experiment through transfer learning [10] using VGGNet, InceptionV3 and XceptionNet.

3 Data Gathering and Preprocessing

We have used BCCD Dataset [11]. The dataset consists of 352 microscopic images of dyed white blood cells. White blood cells (also known as Leukocytes) can be easily distinguished from other types of blood cells due to the presence of nucleus. Each image is of size 640×480 and is in following distribution: 21 Monocyte, 33 Lymphocyte, 207 Neutrophil, 88 Eosinophil, 3 Basophil. The preprocessing was done in three stages: (1) dataset balancing and image augmentation, (2) downsizing image (Tables 1 and 2).

Table 1 Dataset description for multiclass classification

Data label	Training	Testing	Total
Eosinophil	2497	623	3120
Lymphocyte	2483	620	3103
Monocyte	2478	620	3098
Neutrophil	2499	624	3123
Total	9957	2487	12,444

Table 2 Dataset description for binary classification

Data label	Training	Testing	Total
Mononuclear	4996	1247	6243
polynuclear	4961	1240	6201
Total	9957	2487	12,444

3.1 Dataset Balancing and Image Augmentation

Since the original dataset was not balanced and images in each class were not sufficient for training, additional images were derived from the original images using operations such as rotation(the images are rotated by 90°, 180° and 270°), flip (the images are flipped along the horizontal axis and the vertical axis using opencv) and shear (the images are sheared at 30° both vertically and horizontally). The dataset size increased from 352 images to 12,444 images after the application of these operations. Each class had 3100 images approximately, thus balancing the dataset.

3.2 Downsizing Image

The images were original of the size 640 × 480. These were downsized to a size of 120 × 160 in order to reduce the computational time in training as well as testing.

4 Methodology Used

Convolutional neural network [12] or CNN is a special type of deep neural networks that deal with the phenomena such as localization of the receptive field in high volume data, copying of weights forward as well as image sub-sampling using various kernels in each convolution layer. Convolution is a mathematical operation that employs feature extraction over the image.

Pseudocode for seven layer models

```
num_classes = 4
epochs = 20
dropout =0.7
learning_rate = 0.001
batch_size = 32
model = Sequential()
model.add(Lambda(lambda x: x/127.5 − 1., input_shape=(120, 160, 3),
        output_shape=(120, 160, 3)))
model.add(Conv2D(32, (3, 3), input_shape=(120, 160, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dropout(dropout))
model.add(Dense(num_classes))
model.add(Activation('softmax'))
rms = RMSprop(lr = learning_rate,)
model.compile(loss='categorical_crossentropy',
            optimizer= rms,
            metrics=['accuracy'])
```

4.1 Model Architecture

LeNet [13] architecture was used primarily. This section describes in more detail the architecture of LeNet-5, the convolutional NN used in the experiments. LeNet-5 comprises seven layers, not counting the input, all of which contain trainable parameters (weights) [13]. The input is a 120 × 160 pixel image. The pixel values are normalized with respect to 255, and hence, black is associated with a pixel value of 0 and white is associated with a pixel value of 1 (Figs. 1 and 2).

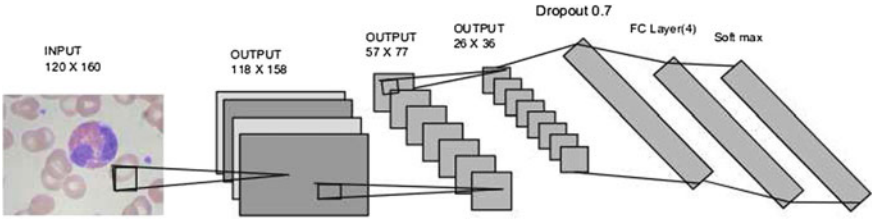
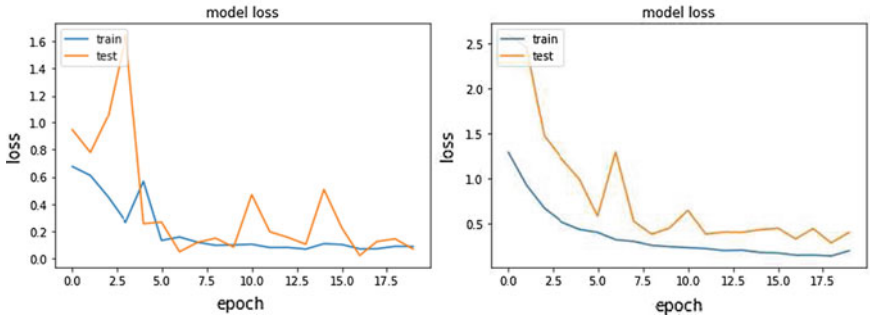


Fig. 1 LeNet-5 CNN structure for a 120 × 160 input image



(a) Binary classification, epoch = 20 Lr = 0.001 Dropout = 0.7

(b) Multiclass classification, epoch = 20 Lr = 0.001 Dropout = 0.7

Fig. 2 Training and testing loss graphs for the best results so obtained in binary and multiclass classification

4.2 Transfer Learning Using VGGNet

VGGNet [14] consists of 16 or 19 convolutional layers and is very appealing because of its very uniform architecture. It only performs 333 times 333 convolutions and 222 times 222 pooling all the way through. The last 10% layers of network were removed and a fully connected layer of four classes for multiclass classification and two classes for binary classification was added and fine-tuned.

4.3 Transfer Learning Using InceptionV3

InceptionV3 [15] gets rid of the linear convolutions that are the bread and butter of CNNs and instead connects convolutional layers through multi-layer perceptrons that can learn nonlinear functions. These perceptrons are mathematically equivalent to 1 × 1 convolutions and thus fit neatly within the CNN framework. InceptionV3 adopts convolution factorization and improved normalization.

4.4 Transfer Learning Using XceptionNet

Xception [16] is an extension of the Inception architecture which replaces the standard Inception modules with depthwise separable convolutions.

5 Experimental Implementation

As part of dataset collection, 352 microscopic images of dyed white blood cells from BCCD Dataset [11] were collected. These files were then rotated, flipped and sheared along with image augmentation to increase the size of dataset to 12,444 images with approximately 3100 images of each white blood cell type. The dataset consists of images of size 640 × 480 which are then resized to 120 × 160 for faster computation. For multiclass classification, 9957 images (2470 of each class approximately) were used for training while 2487 images (620 images of each class approximately) were used for testing purpose. For binary classification, 9957 images (4961 images of each class approximately) were used for training while 2487 (1240 images of each class approximately) were used for testing purpose. LeNet-5 [13] architecture was implemented for both binary and multiclass classifications. String labels of each white blood cell type are encoded using one hot encoding. Binary cross entropy was used for binary class classification of polynuclear and mononuclear white blood cells. Categorical Cross entropy was used for classification into Lymphocyte, Monocyte, Eosinophil or Neutrophil. A batch size of 32 images were used. The models were implemented and executed using the Keras wrappers with Tensorflow framework as the backend to run the models on the Nvidia GTX960M GPU. Each of the models was trained using the RMSprop optimizer, and a suitable dropout was used to maintain a strong resistance to overfitting. The mean squared error function is used for designing the loss function for all the models.

Additionally, Transfer learning using VGGNet [14], InceptionV3 [15] and XceptionNet [16] was performed. The last 10% layers of these models were removed, and a fully connected layer with number of neurons depending upon number of classes was added and only the last layer was trained. This was done to compare results of fine-tuning models with stand-alone Le-Net architecture described above.

6 Results

This section presents the results of the experiment using a LeNet inspired architecture as well as a comparison with transfer learning-based models (Tables 3, 4, 5, 6, 7, 8 and 9).

Table 3 Results for binary classification using CNN architecture

Epochs	Learning rate	Dropout	Accuracy
20	0.001	0.7	**0.963**
20	0.0001	0.7	0.868
50	0.001	0.7	0.940
50	0.0001	0.7	0.959
50	0.0001	0.5	0.902
50	0.0001	0.8	0.924
100	0.0001	0.7	0.937

Table 4 Confusion matrix for binary classification using CNN architecture

Actual\predicted	Mononuclear	Polynuclear
Mononuclear	TP = 1162	FP = 78
Polynuclear	FN = 15	TN = 1232

Table 5 Results for binary classification using transfer learning

Model name	Epochs	Learning rate	Dropout	Accuracy
Inception	20	0.001	0.7	0.7277
Vgg16	20	0.001	0.7	0.6558
Vgg19	20	0.001	0.7	0.5014
Xception	20	0.001	0.7	0.7945

Table 6 Results for multiclass classification using CNN architecture

Epochs	Layers	Learning rate	Dropout	Accuracy
20	7	0.001	0.7	**0.8793**
20	7	0.0001	0.7	0.7414
50	7	0.001	0.7	0.7925
50	7	0.0001	0.7	0.6489
20	11	0.001	0.7	0.8761
20	11	0.0001	0.7	0.7800
50	11	0.001	0.7	0.8484
50	11	0.0001	0.7	0.8383

Table 7 Confusion matrix for multiclass classification using CNN architecture

Actual\predicted	Eosinophil	Monocyte	Lymphocyte	Neutrophil
Eosinophil	518	0	0	105
Monocyte	0	620	0	0
Lymphocyte	0	0	465	155
Neutrophil	39	1	0	584

Table 8 Results for multiclass classification using transfer learning

Model name	Epochs	Learning rate	Dropout	Accuracy
Inception	20	0.001	0.7	0.5761
Vgg16	20	0.001	0.7	0.4306
Vgg19	20	0.001	0.7	0.3884
Xception	20	0.001	0.7	0.6529
Inception	20	0.0001	0.7	0.5597
Vgg16	20	0.0001	0.7	0.3948
Vgg19	20	0.0001	0.7	0.3675
Xception	20	0.0001	0.7	0.6083

Table 9 Comparison with other works

Work	Accuracy
Ongun et al. [5]	0.91
Tai et al. [6]	0.95
Manik et al. [1]	0.99
Chung et al. [2]	0.95
Theera-Umpon [8]	0.77
Our work	0.96

7 Conclusion and Future Work

The current work is aimed at automating the feature extraction and selection process along with the classification of white blood cell. A comparison with transfer learning-based models as well as previous works has been made. Experimental results show that the plain CNN architecture with seven layers shows better results than transfer learning modules as well as other previous semi-automated works. Future work can be done in accommodating medical images in transfer learning architecture to improve results over a large database of several disease images and in a short time. An unbalanced dataset [17] can also be accommodated in future using Breiman's random forest [18].

References

1. S. Manik, L.M. Saini, N. Vadera, Counting and classification of white blood cell using Artificial Neural Network (ANN), in *IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*, IEEE (2016)
2. J. Chung et al., Counting white blood cells from a blood smear using fourier ptychographic microscopy. PloS one **10**(7), e0133489 (2015)
3. M. Habibzadeh, A. Krzyak, T. Fevens, *White Blood Cell Differential Counts Using Convolutional Neural Networks for Low Resolution Images and Soft Computing* (Springer, Berlin, Heidelberg, 2013)
4. M. LaFleur-Brooks, Exploring medical language: a Student-Directed Approach (7th ed.). St. Louis, Missouri, US: Mosby Elsevier. p. 398. ISBN 978-0-323-04950-4 (2008)
5. G. Ongun, et al., An automated differential blood count system. Engineering in Medicine and Biology Society, in *2001 Proceedings of the 23rd Annual International Conference of the IEEE*, vol. 3. IEEE (2001)
6. W.-L. Tai et al., Blood cell image classification based on hierarchical SVM, in *2011 IEEE International Symposium on Multimedia (ISM)*, IEEE (2011)
7. H. Ramoser, Leukocyte segmentation and SVM classification in blood smear images. Mach. Graph. Vis. Int. J. **17**(1), 187–200 (2008)
8. N. Theera-Umpon, S. Dhompongsa, Morphological granulometric features of nucleus in automatic bone marrow white blood cell classification. IEEE Trans. Inf. Technol. Biomed. **11**(3), 353–359 (2007)
9. I. Guyon, A. Elisseeff, *An Introduction to Feature Extraction* (Berlin, Heidelberg, Feature extraction. Springer, 2006), pp. 1–25
10. S.J. Pan, Q. Yang, A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345–1359 (2010)
11. GitHub - Shenggan/BCCD_Dataset: BCCD Dataset is a small-scale dataset for blood cells detection. BCCD Dataset is under MIT licence. [Online]. Available: https://github.com/Shenggan/BCCD_Dataset
12. Krizhevsky, A., I. Sutskever, G.E. Hinton. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Proc. Syst. (2012)
13. Y. LeCun et al., Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
14. K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
15. C. Szegedy, et al. Rethinking the inception architecture for computer vision, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016)
16. J. Carreira, H. Madeira, J.G. Silva, Xception: a technique for the experimental evaluation of dependability in modern computers. IEEE Trans. Softw. Eng. **24**(2), 125–136 (1998)
17. N.V. Chawla, Data Mining for Imbalanced Datasets: An Overview. Data mining and knowledge discovery handbook (Springer, Boston, MA, 2009), pp. 875–886
18. L. Breiman, Random forests. Mach. Learn. **45**(1), 5–32 (2001)
19. I. Goodfellow et al., *Deep Learning*, vol. 1 (MIT press, Cambridge, 2016)
20. Agostinelli, F., et al., Learning activation functions to improve deep neural networks. arXiv preprint arXiv:1412.6830 (2014)