Optimizing AlexNet using Swarm Intelligence for Cervical Cancer Classification

Habibullah Akbar
Computer Science
Esa Unggul University
Jakarta, Indonesia
habibullah.akbar@esaunggul.ac.id

Nizirwan Anwar
Computer Science
Esa Unggul University
Jakarta, Indonesia
nizirwan.anwar@esaunggul.ac.id

Siti Rohajawati
Information System
Bakrie University
Jakarta, Indonesia
siti.rohajawati@bakrie.ac.id

Alivia Yulfitri
Information System
Esa Unggul University
Jakarta, Indonesia
alivia@esaunggul.ac.id

Hafizah Safira Kaurani
Computer Science
Esa Unggul University
Jakarta, Indonesia
hfz.safirakaurani@gmail.com

Abstract — In this study, we optimized a convolutional neural network model i.e. AlexNet to classify images of cervical cancer cells. Although having canonical CNN architecture, AlexNet is only equipped with few hidden layers and thus makes it less efficient for complex objects such as cervical images. To overcome this limitation, we optimized AlexNet using a swarm-based approach (particle swarm optimization). The dataset used is the Intel & MobileODT Cervical Cancer Screening dataset. Firstly, we optimize standard AlexNet based on epoch, data subsets during training (minibatch), learning rate, input image resolution, and training-testing ratio. After having the best parameter values, we derive 3 models of AlexNet based on the number of convolutional layers. Using this approach, AlexNet with a double convolutional layer produces 60.14%, almost as good as the standard residual network on cervical images. However, when AlexNet optimized by swarm-based intelligence (particle swarm optimization) and an additional dropout layer, the accuracy can attain about 67% which is can surpass the standard residual network by 6.22%.

Keywords—AlexNet, particle swarm optimization, medical image processing, cancer classification

I. INTRODUCTION

Cervical cancer is one of the most common types of cancer. Every year, about half a million women worldwide are diagnosed positive for cervical cancer with a mortality rate of more than 300,000 [1]. The high level of prevalence of the incidence rate certainly requires the attention of both the rulers, private sectors, and the communities in efforts to prevent, treat and control cervical cancer. Cervical cancer stems from irregular cell growth and these cells can attack other biological organs either by direct growth in adjacent tissues or by migration to other cells (metastasis). Irregular growth can cause DNA damage, resulting in mutations in vital genes that control cell division, and other organs. If it is severe, cell growth will become a malignant tumour that attacks the tissue in the cervix. The main cause of cervical cancer is Human Papilloma Virus infection, although several other factors can also affect cancer progression.

Generally, cervical cancer tests can be done with a pap test, HPV examination, and IVA tests which are supported by imaging tests / medical images such as CT scans, microscopy, and MRI. However, another potential way of screening is to utilize state-of-the-art computer vision technologies. Recent approaches use neural networks that can learn cell features automatically. This model is known as a convolutional neural network (CNN). CNN can be trained to perform automatic segmentation of cancer cells or detect cancer potential based on medical images taken from patients [2], [3]. However, this CNN technology still requires further research so that it can be developed into a tool for pathology specialists in diagnosing cervical cancer based on patient medical images. Moreover, the performance should be higher than standard human experts or doctors.

In academia, convolutional neural networks (CNNs) has been used to detect the potential of cancer from medical images. A study conducted by [4] showed how a CNN can detect mitosis in cancer cells based on histological images. More in-depth studies regarding the classification of cancer have been carried out by [5]. In addition, [6] has succeeded in using a modified LeNet model with a variety of layers and parameters to reduce the memory and computational load.

CNN can also be used for cervical cancer segmentation without having to do classification [7]. More interestingly, the study conducted by [8] was able to combine the method of segmentation and classification of cancer in one diagnostic framework. In addition, CNN variants such as GoogLeNet and AlexNet have also been utilized for breast cancer classification on histopathological images taken from biopsy samples without involving any segmentation process [9].

Fig. 1 provides a visual illustration of the appearance of cervical cancer cells that appear no more difficult to classify than breast cancer cells. The following is a comparison of breast cancer pictures and cervical cancer pictures.

For breast cancer images, AlexNet was found to have relatively high accuracy i.e. 80% [9]. However, its efficacy on cervical cancer remains unknown. Meanwhile, in a study conducted by [10], the accuracy produced by their proposed CNN deep residual network method did not exceed 60%. Therefore it is necessary to research whether such a CNN model can attain higher accuracy. In this study, we are interested to examine how good AlexNet for classifying

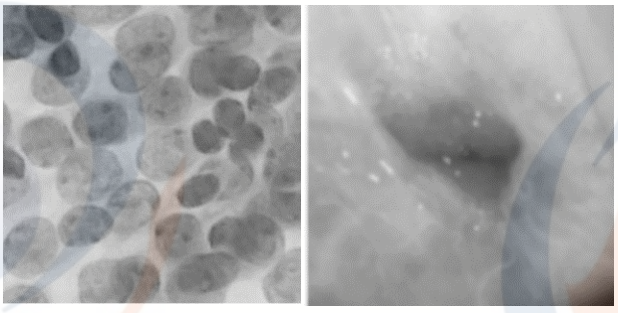

Fig. 1 Image of (left) breast cancer cells [9], (right) cervical cancer cells [10].

978-1-6654-4146-9/21/$31.00 ©2021 IEEE

cervical images. We also have tried to optimize its performance using swarm-based intelligence i.e. particle swarm optimization.

II. METHODS

A. AlexNet's Convolutional Neural Network Architecture

As described previously, AlexNet has the advantage of having only several hidden layers and thus require relatively less computational time for training. Here, AlexNet receives an image as input then processes it through several layers (also called blocks): a convolution layer, a ReLu layer, a pooling layer, and finally a fully-connected layer which classifies cervical input images into 3 types of cancer that based on the softmax activation function.

1) Input patch

Examples of cervical cancer images (type 1, type 2, and type 3) can be seen in Fig. 2.

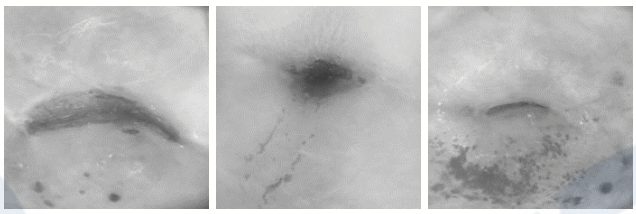

Fig. 2 Training image samples for (a) Type 1, (b) Type 2, and (c) Type 3.

2) Blocks

Each CNN block consists of 3 sub-layers, namely the convolution layer, ReLU, and *pooling*.

- **The convolutional layer** accepts a cervical image as input. This layer consists of a collection of filters that are randomly initialized to look for a feature representation of an image based on the disease type category. The values of the filters are regarded as the network weights that will be optimized during the training phase. Each filter represents a *receptive field* that will detect features from the simplest such as *edges*, *curves* to more complicated features such as *cell parts.* The output from this layer is a filtered image.
- **The ReLU layer determines** whether pixels of the filtered image should be transmitted to the next layer or not. The form of the ReLU function can be expressed by $f(x) = max(0, x)$, which means that it will cut off pixel values less than 0.
- Then, **the *pooling* layer** decreases the size of the image to enable the subsequent blocks to find more features in the lower resolution scale.

3) Fully-connected layers

The full connection layer is similar to the standard network's hidden layer consisting of *perceptron* that responsible classify the previously processed image. We used the standard number of *perceptron* i.e. 4096. Lastly, for the output layer, there are only 3 perceptrons that are associated with the number of cervical types.

In this study, we experiment with three different AlexNet architectures as shown in Fig. 3. These models are derived by varying the number of convolutional layers in each block. AlexNet 1 model has only single convolutional layers while AlexNet 2 and AlexNet 3 models have double and triple convolutional layers respectively.

Fig. 3 The architecture of the 3 AlexNet models based on the variation in the number of convolutional layers in each block (left) AlexNet 1 (middle) AlexNet 2 (right) AlexNet 3.

B. Dataset and Experimental Setup

The dataset used is a standard dataset, namely Intel & MobileODT Cervical Cancer Screening [11]. This dataset contains a total of 1481 images which consist of three label types of cervical cancer as follows:

- Type 1 (consists of 250 images)
- Type 2 (consisting of 781 images)
- Type 3 (consists of 450 images)

Patients with type 1 cervix only need standard screening. Patients with type 2 and type 3 cervix require an advanced screening process [2].

Because this set has an unbalanced number of images between classes or types of cervical cancer, we also created a balanced and smaller version of the dataset based on the image size of the lowest cancer type, which is 250 images. So that the total images for 3 types of cervical cancer become 750 images, 70% of which are used as training data, and the remaining 30% are used as testing data. Originally, the images have resolutions ranging from 480 x 640 to as large as 3096 x 4128. To reduce the computational burden during the training phase, we reduce the image size to 3 resolutions. By default, the experiments use the following training parameters:

- The initial learning rate is 0.0001
- The learning rate drop factor is 0.1
- The L2 Regularization is 0.004
- The minibatch size is 10

The experiments are divided into two main sections. In the first experiment, we manually experiment with the parameters from the epoch, minibatch, and learning rate. Then, we experiment with the training-testing ratio, dataset size, and AlexNet's architectures. For the second experiment, we use PSO to optimize these parameters.

Detail of the first experiment is given as follow. We experiment with AlexNet's sensitivity for the epoch parameter, we set the learning rate to 0.0001 and the mini-batch size to 10. The values of epoch ranging as: 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, and 300. Next, we hold the most optimal epoch value then test other parameters such as minibatch and learning rate. Next, the values of minibatch and learning rate have jointly experimented with the range of (10, 20, 30, 40) and (0.0001, 0.001, 0.01).

Then, we hold the most optimal values of the epoch, minibatch, and learning rate and continue to the next experiment. Here, we compare the influence of the ratio between data training and testing and the effect of the trimmed dataset against the original dataset. Finally, after obtaining the best combination of parameters, tests are carried out on the 3 AlexNet models.

After finding the best model, we continue with the swarm-based experiment. More specifically we design two schemes: (1) the optimization of epoch and minibatch, (2) the optimization of the epoch, minibatch, and additionally parameter i.e. dropout percentage.

The experiments of AlexNet and PSO used the following parameters:

- Population = 5
- Maximum iteration = 10
- The maximum epoch is 15 and the minimum is 1.
- The maximum minibatch is 40 and the minimum is 10.
- The maximum dropout percentage is 100 and the minimum is 0.
- The maximum velocity is 5 and the minimum is -5

The value of the learning rate for both cognitive and social acceleration for updating global best and previous best scores is set as 1.4.

C. Particle Swarm Optimization

To overcome the trial-and-error process that is difficult to do experimentally, the PSO optimization approach is promising. The PSO model is inspired by the swarm behaviour of individuals such as birds and fishes (called particles) which shows a form of intelligence in the social colony (community). The simplification of how birds can coexist with physical movements to find food, mate, and avoid predators has served as an optimization model. In the D-dimensional parameter search space, the position of the particles can be denoted as $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$. Each particle changes its position towards the optimal solution, through the search space, directed by the position of the best particle.

Each particle is modelled with three components, namely momentum, cognitive and social components (Kennedy & Eberhart, 1995). The momentum component is based on the previous velocity. The cognitive component is based on the experience of each particle, represented by the position of the best member pi. The social component is based on the whole swarm experience, represented by the best position *pbest* of the previous member. The best velocity and position that the particles have visited can be represented as $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$ and $p_i = (p_{i1}, p_{i2}, ..., p_{iD})$ respectively.

Then, the position of the next particle can be updated heuristically based on the rules of successive motion as given in (1).

$$x_{i+1} = x_i + v_{i+1} \quad (1)$$

The particle velocity v_{i+1} is a combination of the momentum, cognitive and social components as shown in (2).

$$v_{i+1} = 0.5 \cdot v_i + a_1 \cdot rand_1() \cdot (p_i - x_i) + a_2 \cdot rand_2() \cdot (p_{best} - x_i) \quad (2)$$

The variables a_1 and a_2 are the learning rates of cognitive and social acceleration. The variables $rand_1$, $rand_2$ are random numbers between 0 and 1 with a uniform distribution. The variable velocity of v_i can also be limited to a limit between [$-v_{max}$, v_{max}] while the value of v_{max} can be determined by the number 2 [12][13]. The pseudocode is given in the following.

```
Initialize the population
  Do
    For each particle i = 1 to the N_th member
    if f(x_i) > f(p_i) then p_i = x_i
    if f(p_i) > f(p_best) then p_best = p_i
    For d = 1 to the D-dimension
      v_id = v_id + a_1·rand()·(p_id - x_id) + a_2·rand()·(p_best - x_id)
      v_i = maks (v_min, min (v_max, v_id))
      x_id = x_id + v_id
    next d
    next i
  Until the termination criteria are met
```

D. Performance Evaluation

To measure the performance of the network models, we use the accuracy metric derived from the *confusion matrix* as given in Table I.

TABLE I. CONFUSION MATRIX

Cancer Severity	Prediction		
	Type 1	Type 2	Type 3
Type 1	x_{11}	x_{12}	x_{13}
Type 2	x_{21}	x_{22}	x_{23}
Type 3	x_{31}	x_{32}	x_{33}

The accuracy value can be obtained by comparing the number of correct prediction results against the three types. The total true positive can be calculated using (1).

$$TTP_{all} = \sum_{j=1}^{3} x_{jj} \quad (1)$$

The variable x_{11} is the total true of Type 1, x_{22} is the total true of Type 2, and x_{33} is the total true of Type 3. The overall accuracy A can be calculated using (2).

$$A = \frac{TTP_{all}}{All} \quad (2)$$

All is the total value of all elements in the confusion matrix. In addition to accuracy, we utilize CPU time to measure the length of the training process. Normally, the optimization process (the use of PSO) will extend the training time. The experiments were carried out using the following

specifications. We used Intel (R) Core (TM) i7-6700HQ CPU @ 2.60GHz, 12GB RAM, 1TB HDD, Nvidia GeForce GTX 950 GPU. As for software, we use Matlab 2020b working on Windows 10 Pro 64Bit.

III. Results and Discussion

A. Manually Search Parameter Experiments

Table II shows the results of AlexNet's sensitivity based on the epoch parameter. At epoch 10, the accuracy on AlexNet was only 44.38%. During the training process, the progress succeeded in increasing accuracy, but the validation results were not as high as the training results. At epoch 20, the accuracy of AlexNet was increased to 46.10% and at epoch 30 the accuracy was 47.24%.

TABLE II. Experiments of Epoch

No	Epoch	Accuracy (%)	CPU time (seconds)
1	10	44.38	41
2	20	46.10	74
3	30	47.24	105
4	40	47.43	139
5	50	47.72	113
6	60	48.19	202
7	70	48.57	231
8	80	49.71	265
9	90	49.71	346
10	100	49.52	327
11	150	49.90	486
12	200	49.71	647
13	300	52.19	976

The increase in epoch from 30 to 40 does not seem to affect accuracy (it only increased by 0.19%). Likewise, on epoch 50, accuracy only increased by 0.29%. This can be due to the random number generator factor in the process of initiating the weights of the AlexNet model. The combination of the initial weight values generated can make the model trapped in stagnant accuracy where the gradient value search process cannot find a new combination of weight values that can produce better accuracy.

Epoch 60 managed to increase the accuracy to 48.19%. However, in the next experiment (epoch 70) the accuracy improvement was not significant. Interestingly, epoch 80 managed to increase the accuracy to 49.71. The next experiment was carried out on epoch 90 which neither increase nor decrease the accuracy. The next experiment was carried out at epoch 100, the accuracy decreased to 49.52%. To see the effect of the epoch parameter further, the number of iterations was added to 150 and the result shows that the accuracy goes up to 49.90%. However, when the epoch is added to 200, it returns an accuracy similar to epoch 90.

This shows that a higher epoch does not guarantee the effectiveness of AlexNet. To ascertain the effect of the epoch parameter, we experimented with a value of 300. The result was that the accuracy could reach 52.19%. The time needed to achieve this accuracy is 16 minutes and 16 seconds. For lower epoch e.g. 10, it only requires training time at 41 seconds while the time needed for epoch 80 is 4 minutes 25 seconds.

In general, the increase in epoch has a positive effect as the trend of increasing accuracy. However, between 100 to 200 epochs, accuracy seems to fluctuate so that it has a positive effect on AlexNet's performance. In addition, adding epochs over 300 will further tax the computation time required by the AlexNet model.

Table III shows experiments for minibatch parameters and learning rate. In the minibatch parameter, we tested it with values of 10, 20, 30, and 40. The epoch parameter used is 150 because based on previous experiments this value shows that the AlexNet model has become stable.

The *minibatch* 10 (learning rate 0.0001) gives the best accuracy and CPU *time results* i.e. 49.90% and 486 seconds. Surprisingly, the increase in the value of *minibatch* by 20 and 30 reduces the accuracy with almost the same total time required. However, for *minibatch* 40, the accuracy rate increases again to 46.29% but requires an additional time of 66 seconds. This shows that the minibatch parameter does not have a significant effect on the performance of the CNN AlexNet model. In other words, *minibatch* 10 is good enough for training.

TABLE III. Experiments of Minibatch and Learning Rate

No	Minibatch	learning rate	Accuracy (%)	CPU time (seconds)
1	10	0.0001	49.90	486
2	20	0.0001	46.29	484
3	30	0.0001	37.33	444
4	40	0.0001	46.29	552
5	10	0.001	47.81	616
6	10	0.01	45.90	691

For the *learning rate* parameter, we used the values 0.0001, 0.001, and 0.01. Theoretically, a higher *learning rate* value means the greater the fluctuation in the weight *training* process of the network model. Therefore, we only limit the value to 0.01. The best accuracy value is given at a *learning rate* of 0.0001 i.e. 49.90% with a computation time of 486 seconds and the minibatch value at 10. However, this value also does not have a big effect because it only ranges between 45.90% and 49.90%.

TABLE IV. Experiments of Image Resolution

No	Resolution	Accuracy (%)	CPU *time* (seconds)
1	32 x 32	49.90	486
2	64 x 64	46.10	2621
3	128 x 128	34.67	7382

We have also examined the effect of image resolution on cervical cancer cell input. Table IV describes the effect of resolution parameters on the accuracy and computation time required by the CPU. The other parameters are obtained from previous best experiments: learning rate of 0.0001, minibatch

of 10, and epoch of 150. Surprisingly, this shows that the larger the image size makes the AlexNet model becomes less effective. The computational time at 128 x 128-pixel required 7382 seconds (123 minutes 2 seconds). When compared with the computation time for the input image resolution of 32 x 32, the required CPU time only 486 seconds.

TABLE V. EXPERIMENTS OF DATA TRAINING AND TESTING RATIO

No	Training-testing	Dataset	Accuracy (%)	CPU time (seconds)
1	0.5:0.5 [a] [10]	*trimmed*	45.60	642
2	0.7:0.3	*trimmed*	49.33	593
3	0.8:0.2	*trimmed*	50.00	615
4	0.9:0.1	*trimmed*	50.67	643
5	0.5:0.5 [a] [10]	original	50.41	1316
7	0.7:0.3	original	53.38	1184
8	0.8:0.2	original	56.42	1248
9	0.9:0.1	original	58.11	1637

Table V shows the performance based on the ratio of the *training* and *testing* data used. We set various of the *training* data into 70%, 80%, and 90% and the testing data becomes 30%, 20%, and 10% respectively.

Similar to the previous experiment, we used the best previous values i.e. *epoch* parameters used are 150, the *minibatch is* 10, the *learning rate is* 0.0001, and the input image resolution is 32 x 32 pixels. The *trimmed* dataset is a dataset in which the amount of data in each type has been set to the smallest cancer type category i.e. 250 images. The best accuracy was 58.11% at the training and testing ratio 0.9: 0.1. We also compared it when the ratio used was 50:50 by [10] and it was found that the accuracy was only 50.41%.

After obtaining the optimal value of training and testing data ratios (i.e. 0.9: 0.1), we tested the three AlexNet models. From this test, we wanted to see how the number of convolutional layers on each block affects the accuracy and the computational time. Due to architectural complexity, we used fewer epochs i.e. 15 to compare it to the standard residual network model [10] which has a more complex architecture (i.e., has 32 layers in total).

Table VI shows the performance of the three models. From the experimental results, it was found that the AlexNet 2 was able to achieve an accuracy of 60.14%, which is slightly lower (0.86%) compared to the residual network model which has 32 layers [10]. Interestingly, the AlexNet 3 model which has a total of 21 layers (3 convolution layers for each block) yields the lowest accuracy. This can be due to the insufficient number of training datasets (1481 images) to update the weights on a complex network layer. So that the training results were ineffective. The computation time of the three models is not far apart.

From the experiments that have been carried out, it can be concluded that the best performance of the AlexNet model was given by the AlexNet 2 model (having a total of 18 layers) with 158 seconds for the training time. This model uses epoch 15 parameters, 10 minibatch, 0.0001 learning rate, 32 x 32-pixel input image resolution, and the training and testing ratio is 0.9: 0.1. When compared with research [10], their CNN is the neural network residual model consisting of 32 layers. Even though there are many layers, the accuracy was only 60%. This shows that AlexNet2 was slightly better than the neural network residual model.

B. Swarm-Based AlexNet Optimization Experiments

In this experiment, we optimize the AlexNet using particle swarm optimization (PSO). The letter D represents for dropout parameter while X represents the AlexNet model. The dropout layer is added before the fully connected layer.

Table VII shows that without PSO, the AlexNet 1 model was only attained an accuracy of 51.35% while AlexNet 2 and AlexNet 3 produce 60.14% and 47.97% respectively. However, after optimization with PSO, the AlexNet 1 network accuracy results increased significantly by 7.43% to 58.78%. By adding the dropout layer into this model, the accuracy attains 59.46%. The dropout parameter functions as a regulator which plays a role in helping the network avoid overfitting in the training process. This is in contrast to AlexNet 2 i.e. the PSO-D-AlexNet 2 model produces accuracy 0.68% lower than the AlexNet 2 model.

The dropout parameter which functions as a regulator which plays a role in helping the network avoid overfitting in the training process was also able to increase the accuracy even higher, namely to 59.56%. This is slightly different from the AlexNet 2 model because the PSO-D-AlexNet 2 model gives similar results (0.68% lower than the AlexNet 2 model). However, the PSO-D-AlexNet 3 model is successful in implementing PSO and dropout parameters. gives the best accuracy results i.e. 66.22%.

From the whole experiment, several things can be concluded. The PSO-D-AlexNet 3 model was more accurate than the residual network model [10] which has almost a double layer. The PSO-D-AlexNet 3 model has fewer layers i.e. 22 layers (the AlexNet 3 model consists of 21 layers plus 1 dropout layer). Even though there are many layers, the result of the residual network model accuracy was only 60%, which is 6.22% lower than the PSO-D-AlexNet 3 model

TABLE VI. COMPARISON OF ALEXNET AGAINST RESIDUAL NETWORK MODEL

No	Model	Layers	Accuracy (%)	CPU time (seconds)
1	Residual Network [10]	32	60.00	-
2	AlexNet 1	15	51.35	145
3	AlexNet 2	18	60.14	158
4	AlexNet 3	21	47.97	178

(66.22%). However, this performance is still below the standard of the screening process. This means that the level of accuracy of the cervical cancer classification domain based on deep learning still requires further improvement.

IV. Conclusion

We have evaluated optimized AlexNet using the particle swarm optimization method that has been tested on the Intel & MobileODT Cervical Cancer Screening dataset. For the results without PSO optimization, the AlexNet 2 model produces the best accuracy, which is 60.14%, which is slightly better than the standard residual network model. However, by utilizing PSO and adding a dropout layer, the AlexNet 3 model succeeded in surpassing the residual network model by 6.22%, reaching an accuracy of 66.22%. This model uses epoch 15 parameters, 10 minibatch, 0.0001 learning rate, 32 x 32-pixel input image resolution, and the training and testing ratio is 0.9: 0.1. This shows that AlexNet can classify images of cervical cancer cells better than the standard residual network model. In the future, the performance of the AlexNet model still needs to be improved to be truly applicable in the real health sector. We could also experiment with the others parameters such as weight and learning algorithms or even synthesis augmented dataset.

References

[1] P. A. Cohen, A. Jhingran, A. Oaknin, and L. Denny, "Cervical cancer," *Lancet*, vol. 393, no. 10167, pp. 169–182, 2019.

[2] Z. Liu *et al.*, "Segmentation of organs-at-risk in cervical cancer CT images with a convolutional neural network," *Phys. Medica*, vol. 69, pp. 184–191, 2020.

[3] P. Wang, J. Wang, Y. Li, L. Li, and H. Zhang, "Adaptive Pruning of Transfer Learned Deep Convolutional Neural Network for Classification of Cervical Pap Smear Images," *IEEE Access*, vol. 8, pp. 50674–50683, 2020.

[4] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *International Conference on Medical Image Computing and Computer-assisted Intervention*, 2013, pp. 411–418.

[5] T. Araújo *et al.*, "Classification of breast cancer histology images using convolutional neural networks," *PLoS One*, vol. 12, no. 6, p. e0177544, 2017.

[6] S. Sornapudi *et al.*, "Deep learning nuclei detection in digitized histology images by superpixels," *J. Pathol. Inform.*, vol. 9, 2018.

[7] Y. Song *et al.*, "A deep learning based framework for accurate segmentation of cervical cytoplasm and nuclei," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2014, pp. 2903–2906.

[8] R. Rasti, M. Teshnehlab, and S. L. Phung, "Breast cancer diagnosis in DCE-MRI using mixture ensemble of convolutional neural networks," *Pattern Recognit.*, vol. 72, pp. 381–390, 2017.

[9] M. Żejmo, M. Kowal, J. Korbicz, and R. Monczak, "Classification of breast cancer cytological specimen using convolutional neural network," in *Journal of Physics: Conference Series*, 2017, vol. 783, no. 1, p. 12060.

[10] J. Payette, J. Rachleff, and C. de Graaf, "Intel and MobileODT Cervical Cancer Screening Kaggle Competition: cervix type classification using Deep Learning and image classification," *Stanford University*, 2017.

[11] Intel, "Intel & MobileODT Cervical Cancer Screening Which cancer treatment will be most effective?," 2017. Accessed on: December 01, 2020. [Online]. Available: https://www.kaggle.com/c/intel-mobileodt-cervical-cancer-screening.

[12] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360)*, 1998, pp. 69–73.

[13] H. Akbar, N. Suryana, and S. Sahib, "Training neural networks using clonal selection algorithm and particle swarm optimization: a comparisons for 3D object recognition," In *2011 11th International conference on hybrid intelligent systems*, 2011, pp. 692-697.

TABLE VII. Comparison of Optimized AlexNet Against Residual Network Model

No	Model	Optimum Values			Accuracy (%)	CPU time (seconds)
		epoch	*minibatch*	*dropout*		
1	AlexNet 1	[a] epoch = 15 minibatch = 10			51.35	145
2	AlexNet 2				60.14	158
3	AlexNet 3				47.97	178
4	Residual Network [10]	~15	-	0.4	60.00	-
5	PSO-AlexNet 1	15	25	-	58.78	5931
6	PSO-D-AlexNet 1	11	12	0.65	59.46	4169
7	PSO-D-AlexNet 2	8	26	0.94	59.46	4333
8	**PSO-D-AlexNet 3**	13	31	0.65	**66.22**	5673

[a.] These values are determined manually from the experiments and chosen to be small due to the computational complexity effect from the PSO.