Hyperparameter optimization for deep learning for improved breast mass detection on mammograms

Adarsh Sehgal
Computer Science and Engineering
University of Nevada
Reno, Nevada 89557
Email: asehgal@nevada.unr.edu

Mukul Badhan
Computer Science and Engineering
University of Nevada
Reno, Nevada 89557
Email: mukulbadhan@nevada.unr.edu

***Abstract*—Breast cancer diagnosis through mammography that is accurate has the potential to save millions of lives around the world. In the past, deep learning (DL) has proven to be a godsend for such identification. Any upgrade to the present DL models will be a very useful tool. It's critical to pick the right hyperparameters for such learning models. Genetic algorithms have been shown to be effective in tweaking such parameters in several studies. As a result, we present GA-E2E, a new approach for tuning the hyperparameters of the DL model used to diagnose breast cancer. Our findings reveal that differences in parameter values can considerably alter the area under the curve (AUC), which is used to determine a classifier's performance.**

I. Introduction

The fast growth of machine learning, particularly deep learning, continues to pique the interest of the medical imaging community in using these approaches to increase cancer screening accuracy. Breast cancer is the second-highest cause of cancer death in women in the United States [1], and mammography screening has been shown to lower mortality [2]. Despite its advantages, screening mammography is linked to a high rate of false positives and false negatives. In the United States, the average sensitivity of digital screening mammography is 86.9%, while the average specificity is 88.9% [3]. Since the 1990s, computer-assisted detection and diagnosis (CAD) software [4] has been created to assist radiologists in improving the predicted accuracy of screening mammography. Unfortunately, evidence showed that early commercial CAD systems did not result in significant performance improvements [5]–[7], and that progress remained stagnant for more than a decade after their introduction. With deep learning's remarkable success in visual object recognition and detection, as well as many other domains [8], there is a lot of interest in developing deep learning tools to help radiologists and improve screening mammography accuracy. According to recent studies [9], [10], a deep learning-based CAD system performed as well as radiologists in solo mode, and even increased radiologists' performance in assist mode.

It has been found that when deep learning is used for detecting breast cancer on screening mammography, it can improve the accuracy to as high as 96%. One of the interesting works [11], starts with performing patch classification on a mammogram to find the region of interest (ROI). This patch classifier is then converted to a whole image classifier by introducing a combination of the VGG network [12] and the residual network (Resnet) [13]. [11] uses CBIS-DDSM [14] dataset to perform various experiments to generate an end-to-end training process, which we will refer to as E2E. Depending on the combination of top layers used, the authors have generated various models, which can be transferred to other datasets with no ROI information. Since the learning is being transferred, it needs additional tuning of parameters, as mentioned in [15]. Even though some of the parameters can be tuned manually, it can be a tedious and inefficient task to train the parameters which have a large number of possible values. With each of the parameters assuming a large number of possible values, combinations of these values can grow exponentially. This makes it almost impossible to find near-optimal values for the parameters.

Even though the performance of the whole image classifier to detect a mass in a mammogram has been improved, the authors [15] do not justify the use of many constant parameters used in transferring the learning to a dataset with minimal/no ROI information. [16] and [17] have shown that automatic tuning of such parameters can

greatly enhance the performance of the overall system. [16] made use of GA with Deep Reinforcement Learning (GA-DRL), and [17] used GA with Lidar-monocular visual odometry (GA-LIMO). Both of these works proved that when GA is used for parameter tuning, it has the potential of producing encouraging results.

In this work, we propose a novel algorithm (called GA-E2E), which uses a Genetic Algorithm (GA) on the E2E model to tune the parameters for transferring the learning to Inbreast database [18]. We are using the pre-trained model files from E2E. These pre-trained models were generated for whole-mammogram classification. Each run of a fitness function performs E2E training on the Inbreast dataset using the parameter values decided by the GA. The output of the fitness function is Area Under the Curve (AUC). We are taking GA reference from an open-source implementation [19]. Our tests revealed that GA-E2E is computationally demanding; hence we propose employing a standalone cluster of two PCs connected by a network switch. We used Spark to create this configuration. Our findings demonstrate that GA-E2E contributed to a higher AUC. GA focuses search on the promising portion of the search space, which saves time, computational power, and manual efforts. In the next section, we will describe some closely related works.

II. Background & Motivation

A. End-to-End Pipeline

1) Converting a classifier from recognizing patches to whole images

A common technique for classifying or segmenting large complicated images is to utilize a classifier in sliding window mode to recognize local patches of an image and provide a grid of probabilistic outputs. This is followed by a process that summarizes the outputs of the patch classifier to get the final classification or segmentation result. Using entire slide images of sentinel lymph node biopsies, such approaches have been utilized to detect metastatic breast cancer [20] and to segment neuronal membranes in microscopic images [21]. This technique, however, necessitates two phases, each of which must be tuned separately. Here, training on entire images is performed by combining the two phases into a single step (Fig. 1).

The h function takes complete photos as input and outputs labels for the entire image. As a result, it may be trained from the beginning to the end, giving it two advantages over the two-step technique. First, the entire network may be trained simultaneously, eliminating sub-optimal solutions at each stage; second, the learned network can be transferred to another dataset without relying on ROI annotations explicitly. Large mammography databases with ROI annotations are difficult to come by and very expensive. DDSM, the world's largest public library of ROI annotations for digitized film mammography, contains thousands of pictures with pixel-level annotations that can be used to train a patch classifier. Once the patch classifier has been converted into a whole image classifier h, it can be fine-tuned using only image-level labels from other databases. This method significantly reduces the need for ROI annotations, and it has a wide range of applications in medical imaging, including breast cancer diagnosis on screening mammograms.

2) Network design

A simple way to make an entire picture classifier from a patch classifier is to fatten the heat map and connect it to the image classification output with fully-connected layers. After the heatmap, a max-pooling layer can be employed to boost the model's translational invariance to the patch classifier's output. Furthermore, a shortcut between the heatmap and the output can be created to make training easy. The heatmap is derived straight from the output of the patch classifier, which employs the softmax activation:

$$f(z)_j = \frac{e^{z_j}}{\sum_{i=1}^{c} e^{z_i}} for\ j = 1, ..., c. \tag{1}$$

Convolutional layers, which preserve spatial information, are also employed as top layers. On top of the patch classifier layers, two blocks of convolutional layers (VGG or Resnet) can be added, followed by a global average pooling layer, and finally the image's classification output (Fig. 1). As a result, for full picture classification, this approach provides an "all convolutional" network. The heatmap reduces the depth of the feature map between the patch classifier layers and the top layers, as shown in Fig. 1, potentially causing information loss in the entire picture classification. As a result, when the heatmap was completely removed from the overall image classifier, the upper layers were able to fully utilize the patch classifier's capabilities.

3) Developing patch and whole image classifers on CBIS-DDSM

The pixel-level annotations for the ROIs are stored in the CBIS-DDSM database, along with their pathologically confirmed labels: benign or malignant. Each ROI is also labeled as calcification or mass. The majority of mammograms only had one ROI. All mammograms were converted to PNG format and reduced using interpolation to 1152x896, with no picture cropping. The limitation of GPU memory size prompted the downsizing. By sampling picture patches from ROIs and background regions, two patch datasets were constructed. The size

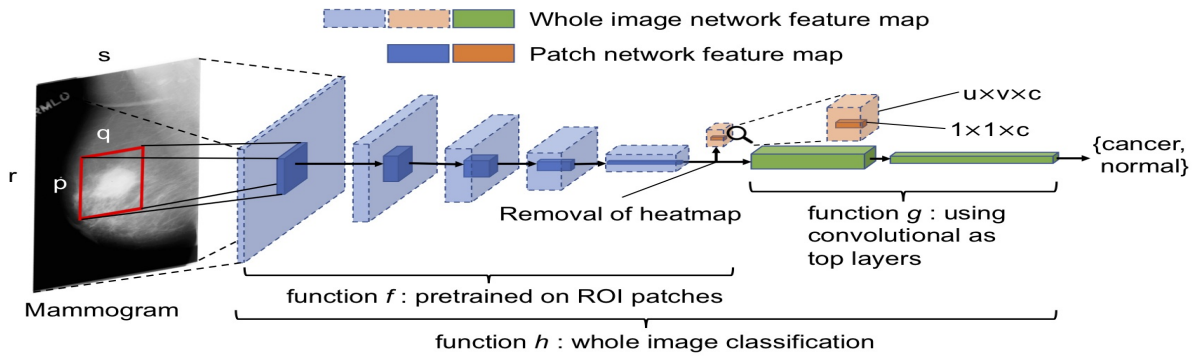

Fig. 1: E2E pipeline.

of all patches was 224x224, which was large enough to cover the majority of the ROIs annotated. The first dataset (S1) was made up of two sets of patches, one of which was centered on the ROI and the other was a random background patch from the same image.

4) Transfer learning for whole image classifcation on Inbreast

The dataset was set up and processed. The Inbreast [18] dataset is a public database that contains FFDM images that were acquired more recently. These images exhibit different intensity profiles than digitized film mammograms from the CBIS-DDSM. As a result, Inbreast is a great way to see if a full picture classifier can be used across different mammography platforms. There are 115 patients and 410 mammograms in the Inbreast database, with both CC and MLO views. Each view is considered a separate image. The radiologists' BI-RADS [22] assessment categories are defined as follows in the Inbreast database: 0: insufficient examination; 1: no findings; 2: benign; 3: probably benign; 4: suspicious; 5: highly indicative of malignancy; 6: known biopsy-proven cancer. [11] provides a platform to perform to transfer the learning to any dataset. In this work, we use [11] on Inbreast, and show that when the parameters are tuned in an optimal way, it can greatly improve the AUC.

B. Genetic Algorithm

GAs [23]–[25] were developed to investigate poorly-understood areas [26], where a comprehensive search is impossible and conventional search methods perform poorly. When GAs are used as function optimizers, their goal is to maximize fitness that is related to the optimization goal. Evolutionary computing approaches in general, and GAs in particular, have had a lot of empirical success on a variety of difficult design and optimization problems. They start with a population of randomly started candidate solutions, which are frequently encoded as a string (chromosome). A selection operator reduces the search space to the most plausible spots, but crossover and mutation operators generate new possibilities.

C. Binary Tournament Selection

A tournament selection (TS) operator is based on a competition among a group of parents. The fitness of a potential solution is calculated for each parent, and the parent with the highest fitness is chosen. The term "binary tournament" refers to a tournament with a field size of two players, which is the most basic kind of tournament selection [27]. The binary tournament selection (BTS) process begins with the selection of two people at random. The fitness levels of these people are then assessed. The person with the best fitness is then chosen. The tournament selection has the advantage of being able to handle either minimization or maximization problems without requiring any structural changes. Furthermore, the use of a negative value is unrestricted [28].

D. Simulated Binary Crossover (SBX)

The search power of the simulated binary crossover (SBX) [29] is similar to that of the single-point crossover. The distinction between real-coded GAs with SBX and binary-coded GAs with single-point crossover implementations is that the former method eliminates variable coding and creates a child string from a probability distribution based on the location of the parent strings.

E. Polynomial Mutation

A polynomial mutation operator with a user-defined index parameter (η_m) was proposed by Deb and Agrawal (1999). They determined from theoretical research that η_m causes a perturbation of $O((b–a)/\eta_m)$ in a variable, where a and b are the variable's lower and upper bounds. They also discovered that in most of the issues they tried, a value of $\eta_m \in [20, 100]$ is sufficient. A polynomial probability distribution is utilized in this operator to perturb a solution in the neighborhood of a parent. The

mutation operator adjusts the probability distribution to the left and right of a variable value so that no value outside the specified range $[a, b]$ is formed. The mutated solution p for a certain variable is constructed for a random number u formed within $[0, 1]$ for a specified parent solution $p \in [a, b]$, as follows:

$$p' = \begin{cases} p + \bar{\delta}_L(p - x_i^{(L)}), & for\ u \leq 0.5, \\ p + \bar{\delta}_R(x_i^{(U)} - p), & for\ u > 0.5. \end{cases} \quad (2)$$

III. Approach

A. Pre-processing Images

Inbreast dataset has images in DICOM format. The dataset comes with a mapping document that maps mammogram image file names to other information such as patient ID, BI-RADS assessment categories, etc. It also provides sample MATLAB code to read the images. We used this code along with our code to split the DICOM images into two classes: positive and negative. We, then, converted these images to PNG format to be used with the E2E algorithm. Only a limited number of converted images were used in our experiments. The images were downsized to 1152×896. Training, validation, and test sub-set splits are described in the following sub-section. Two phases are used in the experiments: GA training, and testing. GA training uses training images to generate a new model built by training the last two layers in E2E. The generated model is then used on validation sub-sets to compute AUC. The output of the fitness function used for GA is the average of AUCs from all epochs. The testing phase is used to compare the GA-found parameters with the original values of the parameters. This phase uses train and validation sub-sets to generate a model, which is then used with the test sub-set to compute the final AUC.

B. Building Spark cluster

This study uses Spark [30] to run GA in a distributed setup. This is important because each run of the fitness function is computationally intensive. We used two computers to build a cluster. Advantages of cluster setup include scalability and the ability for a program to finish in the real possible time. We describe the details of the GA-E2E algorithm in the next sub-section. The word *cluster* is used to describe the setup of two or more computers running in parallel to achieve a common goal. We chose Spark for our work because it provides easy-to-use configurations, and online resources/documentation. The high-level architecture of Spark is shown in figure 2. A cluster manager connects *SparkContext* and *Worker Nodes*. For our work, the cluster was built as in figure 3. The details of the corresponding computers are shown in figure 4. The computers are connected via a network switch. For our implementation specifically, we ran one worker on each of the machines.

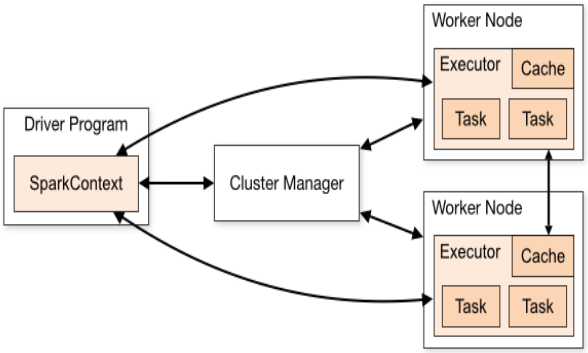

Fig. 2: Architecture of Spark cluster.

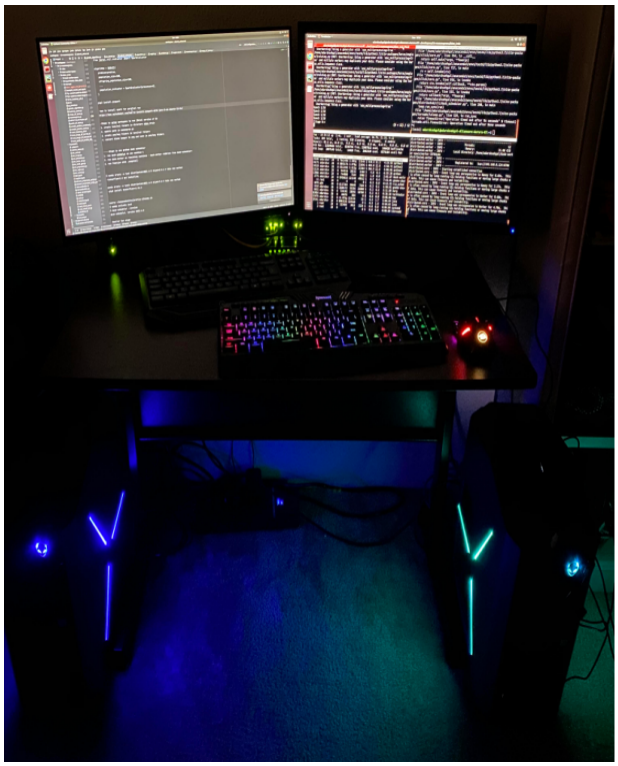

Fig. 3: An actual image of computers setup with Spark cluster.

C. GA-E2E algorithm

Now, we set up GA from [19]. As described in the background section, GA uses BTS, SBX crossover, and polynomial mutation. Since the total training epochs are set at ten, the approximate time for each fitness function evaluation is about 70 seconds for each of the workers.

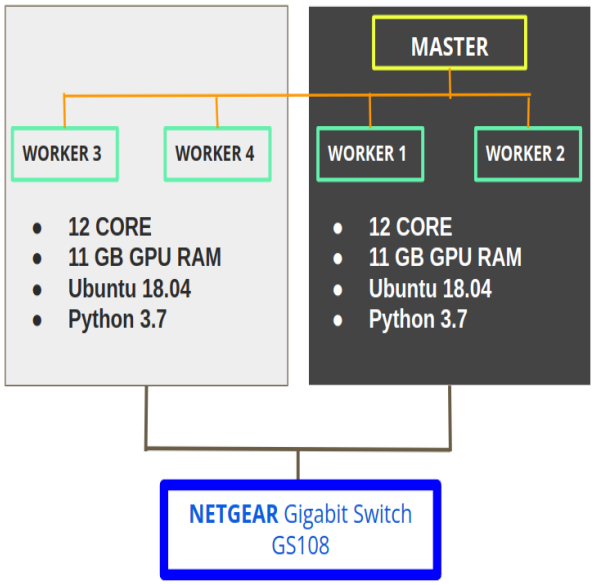

Fig. 4: Example representation of master and worker setup for figure 3.

Considering this, we have set the population size to 70, to be run for 70 generations. GA was run on the model from [15] using transfer learning on the Inbreast dataset. For each evaluation of the fitness function, GA selects values of parameters, uses them to train the network, computes average validation AUC, and finally uses this trained model on the test sub-set. The AUC returned from the fitness function is the AUC computed on the test sub-set. GA training phases use 30 images in the training sub-set, and 12 images in the validation sub-set. Algorithm 1 describes the high-level overview of the GA-E2E algorithm.

Algorithm 1 Proposed GA-E2E Algorithm

1: Choose population of n chromosomes
2: Set the values of parameters into the chromosome
3: **for** all chromosome values **do**
4: Run the code to fine tune the existing model using independent dataset (Inbreast)
5: **return** AUC
6: **end for**
7: Perform SBX Crossover
8: Perform Polynomial Mutation
9: Repeat for required number of generations to find optimal solution

D. Evaluation

The experiments are broken into two sections, as previously stated: GA training and testing. In both phases, the training and validation sets are identical. However, the final AUC is computed using 100 mammogram images in the testing phase.

IV. Experimental Results

A. Experimental Setup

It is important to plan the experiments to be executed to test the algorithm, some details of which we have described in the previous sections. The overall algorithm is described in 1. The workspace was set up with [15]. This code link provides further links to the pre-trained models on whole image classification using the CBIS-DDSM dataset.

Spark requires that the master and worker be able to communicate key-less using SSH. So, we established password-less SSH between our machines. Both machines can SSH as a user/root into other machines. Both computers have the same username, folder structure, GA-E2E code, same images used in train/validation sub-sets, and the .jar files used by Spark. The number of parallel processes was set to two, one for each worker.

Table I shows the parameters we used in GA-E2E algorithm. Each set of parameters decided by the GA was used to perform transfer learning on the Inbreast dataset. The reward for the fitness function is the validation AUC averaged over epochs.

Parameter	Description
pos-cls-weight	weight of positive class
neg-cls-weight	weight of negative class
weight-decay	stage 1
weight-decay2	stage 2
init-learningrate	learning rate for stage 1
all-layer-multiplier	multiplier for stages other than stage 1

TABLE I: Parameters used in the GA-E2E, along with their description.

Each of the parameters is binary encoded into a single chromosome, an example of which is shown in figure 5.

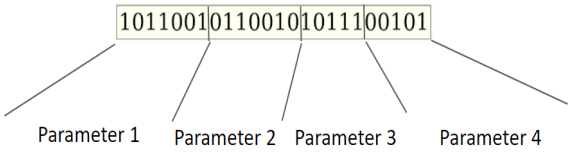

Fig. 5: Chromosome representation for the GA.

From the various manual experiments we did, we would like to mention a few of them. In one such experiment, we used 8 images in a training sub-set, out of which four were negative, and four were positive. By positive we mean that the mammogram image of the breast is malignant, and by negative we mean that the mammogram image of the breast is benign. A similar

setup was done for the test sub-set. For the validation sub-set, two images each of the positive, and negative classes were used. Training epochs were fixed to four, and the number of cores of the GPU to be used was fixed to six. The sub-sets chosen had multiple views (CC/MLO) of the same patient. Each epoch took about 160 seconds to complete. On using the learned model on a test sub-set, we got the AUC as 0.5. We experimented with a various number of images in different sub-sets and found that the program ran for a very long time when more images were used. Since GA needs to run the training a few hundred times to find the near-optimal parameter values, for the rest of our implementation, we decided to use the number of images as in the sub-set division described in this section. However, to run GA-E2E, we used only CC views from different patients for training, validation, and testing.

Finally, we used 30 images for training, 12 for validation, and 100 for test sub-sets. Each GA fitness function was run for ten epochs, while the testing phase had 100 epochs. The E2E uses an early stopping approach; hence the training stopped between 30-50 epochs each time the training was run. This was applicable for both original and GA-found parameters. Our experiment assumes each view of the mammogram as a separate image.

B. Running GA-E2E

The system has now been set up to run the algorithm GA-E2E from 1. Based on the previously described system setup, the GA completed running in about two days. We captured the logs of each run of the fitness function and the corresponding values for each of the parameters. Table II shows the comparison of values of parameters for E2E and GA-E2E. E2E parameters are the ones originally used in transfer learning with Inbreast in [15]. It can be observed that changing parameter values have a significant effect on the AUC of the test sub-set. The increase in AUC is as high as 11% when compared the to original AUC.

Parameter	Range	E2E, AUC=0.51	GA-E2E AUC=0.57
pos-cls-weight	0 - 0.999	1.0	0.948
neg-cls-weight	0 - 0.999	1.0	0.319
weight-decay	0 - 0.999	0.0001	0.269
weight-decay2	0 - 0.999	0.0001	0.225
init-learningrate	0 - 0.999	0.01	0.0008
all-layer-multiplier	0 - 0.999	0.1	0.475

TABLE II: E2E vs. GA-E2E values of parameters.

C. Training evaluation

It is significantly important to monitor the progress of a GA as it is running to optimize the performance of the system. From the way GA's work, some of the chromosomes will outperform others. Ultimately, the average fitness would increase towards the maximum fitness value. Figure 6 shows the best and average fitness value (AUC) over generations.

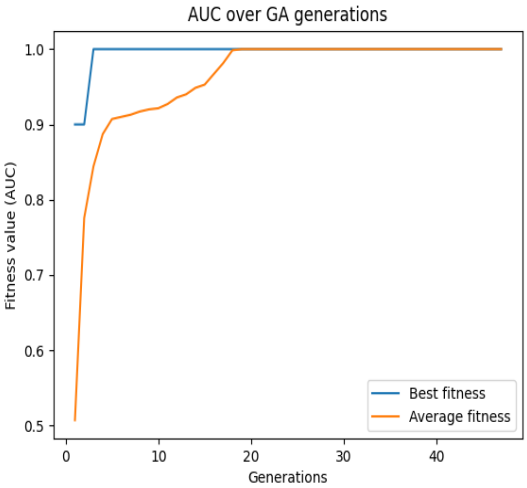

Fig. 6: Fitness value evaluation over GA generations.

D. Analysis

It can be observed that all generations except the first two, saw the best fitness value of 1.0. However, the average fitness value started at 0.5, and ultimately, the GA was able to have the entire population at 1.0 AUC. GA took about 18 generations to reach the maximum success rate. Even though the GA reached its maximum success rate, the system was left to run until completion.

V. Experimental Issues

Configuration of the Spark cluster posed major challenges. Each configuration parameter needs to be appropriately set for the standalone cluster to work using GPUs. The GPU makes program execution much faster, and running the program on a cluster of two computers, cuts the running time to half. Any combination of incorrectly set-up configuration values can result in an unequal load on the machines, which can further lead to some machines being under-utilized/over-utilized.

VI. Conclusion and Future Work

A. Conclusion

This paper shows that there is no pattern between the values of parameters and the AUC; hence the role of GA

is significant in tuning the parameter values. We also show that GA focuses search in the promising search space. GA started with an AUC close to zero. As the GA progressed, the AUC got higher and ultimately surpassed the originally achieved AUC of 0.51. This points to an increase of around 11% in comparison to the original AUC.

B. Future Work

We believe that GA may be able to reach the AUC of 1.0 much earlier if GA operators are designed specifically for E2E. Also, different convolution layers can be tested with E2E for better adaptability of the learned model with an independent dataset. Adding more computers to the cluster may reduce execution time further. Modified GA-E2E could be run on a supercomputer, which would enable the ability to perform various experiments using a variety of datasets.

REFERENCES

[1] R. A. Smith, D. Saslow, K. A. Sawyer, W. Burke, M. E. Costanza, W. P. Evans III, R. S. Foster Jr, E. Hendrick, H. J. Eyre, and S. Sener, "American cancer society guidelines for breast cancer screening: update 2003," *CA: a cancer journal for clinicians*, vol. 53, no. 3, pp. 141–169, 2003.

[2] K. C. Oeffinger, E. T. Fontham, R. Etzioni, A. Herzig, J. S. Michaelson, Y.-C. T. Shih, L. C. Walter, T. R. Church, C. R. Flowers, S. J. LaMonte *et al.*, "Breast cancer screening for women at average risk: 2015 guideline update from the american cancer society," *Jama*, vol. 314, no. 15, pp. 1599–1614, 2015.

[3] C. D. Lehman, R. F. Arao, B. L. Sprague, J. M. Lee, D. S. Buist, K. Kerlikowske, L. M. Henderson, T. Onega, A. N. Tosteson, G. H. Rauscher *et al.*, "National performance benchmarks for modern screening digital mammography: update from the breast cancer surveillance consortium," *Radiology*, vol. 283, no. 1, pp. 49–58, 2017.

[4] M. Elter and A. Horsch, "Cadx of mammographic masses and clustered microcalcifications: a review," *Medical physics*, vol. 36, no. 6Part1, pp. 2052–2068, 2009.

[5] J. J. Fenton, S. H. Taplin, P. A. Carney, L. Abraham, E. A. Sickles, C. D'Orsi, E. A. Berns, G. Cutter, R. E. Hendrick, W. E. Barlow *et al.*, "Influence of computer-aided detection on performance of screening mammography," *New England Journal of Medicine*, vol. 356, no. 14, pp. 1399–1409, 2007.

[6] E. B. Cole, Z. Zhang, H. S. Marques, R. E. Hendrick, M. J. Yaffe, and E. D. Pisano, "Impact of computer-aided detection systems on radiologist accuracy with digital mammography," *AJR. American journal of roentgenology*, vol. 203, no. 4, p. 909, 2014.

[7] C. D. Lehman, R. D. Wellman, D. S. Buist, K. Kerlikowske, A. N. Tosteson, D. L. Miglioretti, B. C. S. Consortium *et al.*, "Diagnostic accuracy of digital screening mammography with and without computer-aided detection," *JAMA internal medicine*, vol. 175, no. 11, pp. 1828–1837, 2015.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[9] A. Rodriguez-Ruiz, K. Lång, A. Gubern-Merida, M. Broeders, G. Gennaro, P. Clauser, T. H. Helbich, M. Chevalier, T. Tan, T. Mertelmeier *et al.*, "Stand-alone artificial intelligence for breast cancer detection in mammography: comparison with 101 radiologists," *JNCI: Journal of the National Cancer Institute*, vol. 111, no. 9, pp. 916–922, 2019.

[10] A. Rodríguez-Ruiz, E. Krupinski, J.-J. Mordang, K. Schilling, S. H. Heywang-Köbrunner, I. Sechopoulos, and R. M. Mann, "Detection of breast cancer with mammography: effect of an artificial intelligence support system," *Radiology*, vol. 290, no. 2, pp. 305–314, 2019.

[11] L. Shen, L. R. Margolies, J. H. Rothstein, E. Fluder, R. McBride, and W. Sieh, "Deep learning to improve breast cancer detection on screening mammography," *Scientific reports*, vol. 9, no. 1, pp. 1–12, 2019.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[14] R. S. Lee, F. Gimenez, A. Hoogi, K. K. Miyake, M. Gorovoy, and D. L. Rubin, "A curated mammography data set for use in computer-aided detection and diagnosis research," *Scientific data*, vol. 4, no. 1, pp. 1–9, 2017.

[15] "https://github.com/lishen/end2end-all-conv," 2019.

[16] A. Sehgal, H. La, S. Louis, and H. Nguyen, "Deep reinforcement learning using genetic algorithm for parameter optimization," in *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2019, pp. 596–601.

[17] A. Sehgal, A. Singandhupe, H. M. La, A. Tavakkoli, and S. J. Louis, "Lidar-monocular visual odometry with genetic algorithm for parameter optimization," in *International Symposium on Visual Computing*. Springer, 2019, pp. 358–370.

[18] I. C. Moreira, I. Amaral, I. Domingues, A. Cardoso, M. J. Cardoso, and J. S. Cardoso, "Inbreast: toward a full-field digital mammographic database," *Academic radiology*, vol. 19, no. 2, pp. 236–248, 2012.

[19] A. Benitez-Hidalgo, A. J. Nebro, J. Garcia-Nieto, I. Oregi, and J. Del Ser, "jmetalpy: A python framework for multi-objective optimization with metaheuristics," *Swarm and Evolutionary Computation*, vol. 51, p. 100598, 2019.

[20] D. Wang, A. Khosla, R. Gargeya, H. Irshad, and A. H. Beck, "Deep learning for identifying metastatic breast cancer," *arXiv preprint arXiv:1606.05718*, 2016.

[21] D. Ciresan, A. Giusti, L. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," *Advances in neural information processing systems*, vol. 25, 2012.

[22] J. Timmers, H. van Doorne-Nagtegaal, H. Zonderland, H. Van Tinteren, O. Visser, A. Verbeek, G. Den Heeten, and M. Broeders, "The breast imaging reporting and data system (bi-rads) in the dutch breast cancer screening programme: its role as an assessment and stratification tool," *European radiology*, vol. 22, no. 8, pp. 1717–1723, 2012.

[23] L. Davis, "Handbook of genetic algorithms," 1991.

[24] J. H. Holland, "Genetic algorithms," *Scientific american*, vol. 267, no. 1, pp. 66–73, 1992.

[25] D. E. Goldberg and J. H. Holland, "Genetic algorithms and machine learning," *Machine learning*, vol. 3, no. 2, pp. 95–99, 1988.

[26] K. De Jong, "Learning with genetic algorithms: An overview," *Machine learning*, vol. 3, no. 2-3, pp. 121–138, 1988.

[27] T. Bäck, D. B. Fogel, and Z. Michalewicz, *Evolutionary computation 1: Basic algorithms and operators*. CRC press, 2018.

[28] M. Wahde, *Biologically inspired optimization methods: an introduction*. WIT press, 2008.

[29] K. Deb, R. B. Agrawal *et al.*, "Simulated binary crossover for continuous search space," *Complex systems*, vol. 9, no. 2, pp. 115–148, 1995.

[30] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10)*, 2010.