Using ABC Algorithm as a Hyperheuristic for Optimal UNet Hyperparameter Search

Yue Zhao, Kylel Scott
Arizona State University, yzhao326@asu.edu
Arizona State University, kdscott9@asu.edu

KEYWORDS: U-Net, Autoencoder, Artificial Bee Colony, ABC Optimization, hyperheuristic, Semantic Segmentation

I. INTRODUCTION

Autoencoders have been deployed to a number of image segmentation tasks for their ability to learn associations in images through latent feature comprehension [1][7]. However, autoencoders and other complex deep neural network architectures are difficult to fine-tune and resource intensive to train given their complexity and over-parameterization and tendencies to issues like vanishing gradient. Neural networks are seen as metaheuristic to solve non-convex optimization problems, however they can be observed macroscopically as a separate optimization problem, of which a solution can be found that exhibits optimal parameters for the network to perform at optimal accuracy. This work implemented Artificial Bee Colony Optimization (ABC) to find optimal architecture and training parameters for a Deep Autoencoder Network trained for an image (semantic) segmentation task. The team employed U-Net, a fully convolutional network with successive contracting and expansive layers and concatenation between same resolution contracting and expansive layers [1]. The U-Net model was trained in the experiments using the Oxford-IIIT Pets dataset. The ABC algorithm is able to converge on a set of parameters that maximized the accuracy of the U-Net model, and increased the accuracy of semantic segmentation output.

II. RELATED WORK

The Artificial Bee Colony (ABC) algorithm is a global optimization algorithm based on swarm intelligence proposed by Karaboga in 2005. It is inspired by the honey gathering behavior of bee colonies, in which bees perform different activities according to their division of labor and share and exchange information of the colony to find the optimal solution of the problem.

A. ABC Algorithmn Principle

This algorithm takes its inspiration from honeybees, a swarm animal who seeks to find quality nectar sources with extreme efficiency in any environment and is able to adapt to environmental changes. The Honey Collection system in a bee colony consists mainly of food sources, employed bees, onlooker bees, and scout bees. Modeling the natural procedure of honeybee's Honey Collection system as an optimization problem, the food source is observed as a feasible solution to the optimization problem in question. The quality of this food source, its merit, is evaluated by fitness. The employed bees use a greedy criterion to compare the optimal solution in memory with the domain search solution, keeping the best solution. After all the employed bees finish the neighborhood search, the employed bees go back to the hive and share the honey source information with the onlooker bees. The onlooker bees are selected with a certain probability based on the food source information of the employed bees while searching the neighborhood near the source. When the solution is trapped in a local optimum somewhere, the scout bee uses a large-step search to explore the optimal solution.

B. ABC Algorithmn Explanation

The complete steps of the ABC Algorithm are in Appendix A. An overview of the main steps that dictate the progression of the algorithm are explained below.

a. Food Source Initialization

First, the number of food sources is defined as $nPop$, and the quality of the food source (which is proportional to its relative attractiveness) is equivalent to its evaluation with respect to the optimization function; in heuristic algorithms this is referred to as the fitness value. The location of the food sources is defined according to Equation (1). The variables L_d and U_d denote the upper and lower bounds of the traversal, respectively.

$$x_{id} = L_d + rand(0,1)(U_d - L_d) \tag{1}$$

b. Update of new food sources

The employed bee searches for a new nectar source around nectar source i according to Equation (2), where φ is a [-1,1] uniformly distributed random number that determines the degree of perturbation. The a is the acceleration factor (usually taken as 1). When a new nectar source is found that is better adapted, the better one is chosen to replace the original.

$$x_{id}^{new} = x_{id} + a * \varphi\left(x_{id} - x_{jd}\right); \quad j \neq i \tag{2}$$

c. Probability of onlooker bee following an employed bee

The probability of the onlooker bee picking the employed bee follow the Roulette rules, and is described in Equation (3) as p_i.

$$p_i = \frac{f_i t_i}{\sum_{i=1}^{nPop} f_i t_i} \tag{3}$$

d. Generate scout bees

During the search, if the nectar source is not updated to a better nectar source after consecutive iterations of the search and reaches, and the enumeration of previous iterations reaches a threshold L, the nectar source is abandoned. In this event, the scout bee seeks to find a new nectar source on the swarm's behalf, and the location of the nectar source found by the scout bee is explained by Equation (4).

$$x_i = \{L_d + rand(0,1)\left(U_d - L_d\right) \tag{4}$$

$$trial \geq L\, x_i;\ trial < L$$

C. U-Net and Semantic Segmentation

Image Segmentation is the process of segmenting an image with respect to a group of pixels, associating specific pixels with other pixels that represent the same object. Semantic Segmentation is a type of image segmentation that focuses on grouping all similar instances under one group, and is tailored specifically for singular instances of different groups. It can be seen as a pixel-wise analogue to Classification and Localization (as shown in Figure 1), and is commonly used for more refined image analyses that rely on precise comprehension of object boundaries like medical imaging, robot autonomy, and even textured surfaces [1][6][7].

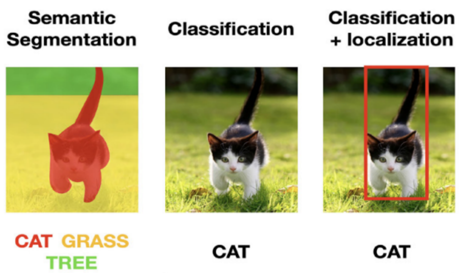

Figure 1: Comparison of Semantic Segmentation, Image Classification, and Image Localization

U-Net is a deep learning architecture that has been observed to excel at the image segmentation task [1][7]. At the time, it outperformed the commonly implemented architecture for image segmentation, being Fully Convolutional Networks. U-Net is an encoder-decoder network with skip connections between identical resolution contracting-expansive layers. The network takes input images and contracts them to a smaller resolution with a wider feature space to progressively extract latent features, then expands them to the original feature space and resolution. In the expansive layer, the latent features at each resolution on the contraction side are shared with the corresponding expansive path through skip connections. The extraction of latent features, iterative refinement of low-resolution feature maps, and sequential concatenation of latent-space data between contracting and expansive paths allow the model to generate precise segmentation maps much more accurately than regular FCNs. The inclusion of skip connections also allow for more efficient back-propagation of the network's optimizer, mitigating the vanishing gradient problem, and has been shown to change the loss landscape as shown in Figure 3 [8].

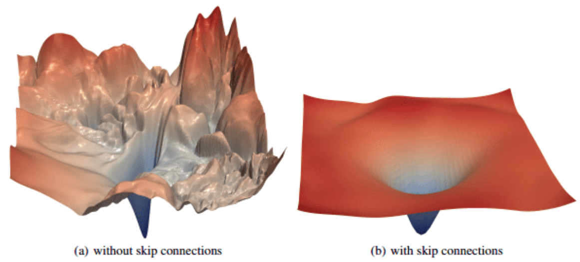

Figure 2: Depiction of Loss landscape of ResNet-56 with and without skip connections

V. IMPLEMENTATION In the U-net model, hyperparameters were required to be given. The main hyperparameters are: learning rate, epochs, dropout rate, batch size and pooling type. The potential values of these dimensions varied in type (numeric and categorical) and bounds. This made the optimization across this feature space a prime opportunity for ABC implementation. A suitable combination of hyperparameters was sought after for a proper model to obtain optimal accuracy. By iteratively and intelligently tuning the hyperparameters of U-Net through ABC optimization, the predicted segmentation masks of the U-Net model became more accurate.

The feature space for ABC optimization was multi-dimensional, however the optimization function was singular: segmentation accuracy. It follows that this problem is converted into a multimodal optimization problem with a single objective. Remarkably, the artificial bee colony algorithm is exactly a multimodal global optimization method. Moreover, this algorithm itself is not very computationally intensive, which may be a promising optimization strategy for large computationally intensive things like training U-Net.

The detailed steps of how ABC was implemented to optimize the hyperparameters of U-Net can be seen in Appendix A.

VI. RESULTS Table 1, shown in Appendix A, shows the optimal combination of the observed hyperparameters for each iteration.

Figure 3 shows the progressive improvement of U-Net segmentation output as a function of the ABC optimization succession.

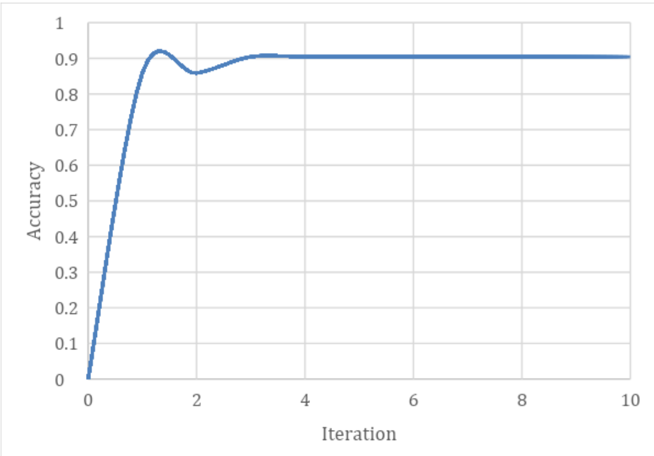

Figure 3: Plot showing progression of each iteration of ABC and accuracy of optimal solution

Figure 4 shows the output of the model trained with the optimal parameters found in the final iteration.

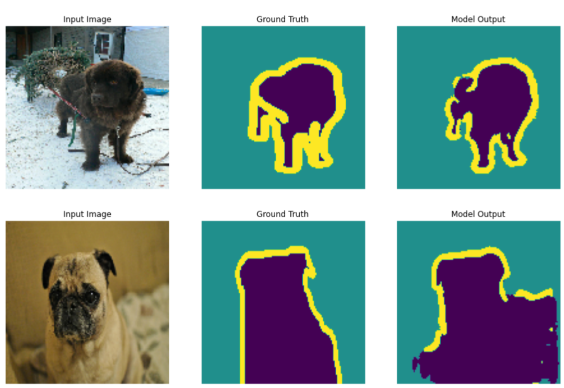

Figure 5: Segmentation output of Image samples from testing data (left: Input Image, middle: Ground Truth Segmentation Mask, right: Model Output)

VII. CONCLUSION From the experimental results, it is concluded that smaller learning rate, larger epochs, and larger dropout rate will make the model more accurate. The optimal batch size is stable at 64, and Max Pooling was shown to be a better pooling type than Average Pooling. With the optimal hyperparameter set (found at iteration 4 out of 10) the accuracy converges to 0.904.

With several iterations, the accuracy tends to increase until it stabilizes. The increase is rapid because we choose a small epoch range. Since there are few combinations of parameters as candidates, the artificial bee colony algorithm can easily find the optimal solution. The onlooker bee in the artificial bee colony algorithm can find a better solution in the neighborhood of the nectar source where the employed bee is located, converging the iterative results toward the optimal value. The scout bees can avoid the local optimum by skipping the neighborhood where the optimal solution is available and searching without being bound by the existing results.

VII. DISCUSSION The results of the experiment would be more intuitive if the artificial bee colony algorithm could be implemented on a wider value domain for the optimization of U-net. Much of the parameters were either binary categorical (Pooling type) or seemed to have an insignificant effect on training accuracy. This is proven between trials 2 and 3; the highest change in accuracy happened when the Batch Size halved and the Pooling Type changed, however all other hyperparameters exhibited minute changes showing their effect was minimal.

Unfortunately, the experimental procedures were limited as a function of computational resources. Training each network took a notable amount of time, and additional iterations of ABC proved to be too costly with the time left in class. Variance in U-Net hyperparameter search could have been improved with the incorporation of a Validation set for the testing procedure. The tradeoff between accuracy and computation time must be considered due to computational power limitations in most cases. The optimized hyperparameters can also be obtained by using validation datasets, which means training on a small amount of data.

VI. REFERENCES

[1] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In MICCAI, pages 234–241. Springer, 2015

[2] Kumar A.,Kumar D. and Jarial S. (2017) A Review on Artificial Bee Colony Algorithms and Their Applications to Data Clustering. Cybernetics and Information Technologies, Vol.17 (Issue 3), pp. 3-28.

[3] Karaboga, D. and Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization, 39(3), pp.459-471.

[4] Karaboga, D. and Akay, B., 2009. A comparative study of artificial bee colony algorithm. Applied mathematics and computation, 214(1), pp.108-132.

[5] Wang, S.G. and Jiang, S., 2022. Optimal Hyperparameters and Structure Setting of Multi-Objective Robust CNN Systems via Generalized Taguchi Method and Objective Vector Norm. arXiv preprint arXiv:2202.04567.

[6] https://ieeexplore.ieee.org/abstract/document/8820616

[7] https://www.mdpi.com/2076-3417/9/18/3900

[8] https://arxiv.org/abs/1712.09913

VII. APPENDIX A.

1. **ABC Algorithm Steps**
   1.1. Generate initial honey sources and determine their location; and initialize the algorithm parameters
   1.2. The employed bee searches for a new nectar source and replaces it if it is a better one.
   1.3. Calculate the probability of the employed bee being followed.
   1.4. Select employed bee according to roulette rules and search for a new nectar source, or substitute if better.
   1.5. If a nectar source does not find a better one within the threshold L, it is abandoned. The scout bee is left to find a better nectar source.
   1.6. Keep iterating until $maxgen$ iterations have passed

2. **Steps for ABC Hyperheuristic Implementation**
   2.1. Randomly generate multiple sets of initial hyperparameters as employed bees and find the initial accuracy of each combination as a nectar source.
   2.2. The employed bees search for a new honey source, which means it iterates through all hyperparameter combinations according to Equation (2) to generate a new combination, and analyzes whether the parameters in the new combination are out of bounds, and regenerates them if they are out of bounds. If the accuracy of the new combination is higher than that of the original combination with the same index, replace it.
   2.3. Calculate the cumulative probability, which is the probability that the employed bee will be followed.
   2.4. Select the employed bee according to the roulette wheel rule, the selected combination is the onlooker bee, and generate a new combination of hyperparameters randomly. And calculate the new accuracy, which is the nectar source. The original combination would be replaced if the accuracy is better.
   2.5. According to Equation (4), the combination is discarded if a better accuracy is not found within the threshold L. Regenerate the combination as a scout bee to find a better nectar source.
   2.6. Continue iterating until a suitable combination of hyperparameters is found.

Table 1: ABC Progression and Optimal Solutions at Each Iteration

Iteration	Learning Rate	Epochs	Drop out Rate	Batch Size	Pooling Type	Accuracy
1	0.001	10	0.1	128	AP	0.85878
2	0.001	10	0.1	128	AP	0.85878
3	0.001	10	0.0939	64	MP	0.90419
4	0.001	10	0.093866	64	MP	0.904196
5	0.001	10	0.093866	64	MP	0.904196
6	0.001	10	0.093866	64	MP	0.904196
7	0.001	10	0.093866	64	MP	0.904196
8	0.001	10	0.093866	64	MP	0.904196
9	0.001	10	0.093866	64	MP	0.904196
10	0.001	10	0.093866	64	MP	0.904196