Neural Architecture Search Using Harmony Search Applied To Malaria Detection

Leonardo Nakatani Moretti and Leandro Dos Santos Coelho

Abstract Over 200 million malaria cases worldwide lead to half a million deaths annually. Although significant progress has been made, erradication remains elusive. One of the main challenges to overcome is diagnosis. Currently, there are many techniques available, among them Light Microscopy being the golden standard. However, this method is slow and expensive, since it requires a professional microscopist, manunally counting Red Blood Cells (RBCs). Some automation attemps have been made, but thus far, no commercial solution has been developed, and there is no consensus on the right approach to the matter. One remakable prospect is the use Convolutional Neural Networks (CNNs) to classify and count RBCs. This method has proven to be highly accurate, but computationally intensive. This work seeks to find more cost-effective topologies, while still maintaining reasonable accuracy. To do so, it will use Harmony Search , a music-inspired metaheuristic. The main contributions of this work is the search of more efficient topologies and the suggestion of a new metric for kind of topology implementation, one that utilizes both accuracy and computation cost indicators.

Key words: Malaria, Harmony Search, Convolutional Neural Network

1 The Malaria Detection Problem

Malaria has afflicted society for a long time. It causes half a million deaths every year. The World Health Organization has malaria erradication in its objectives for

Leonardo Nakatani Moretti
ELetrical Engineering Graduate Program(PPGEE) Federal University of Paraná (UFPR), Curitiba, PR, Brazil e-mail: lnakatani@yahoo.com.br

Leandro Dos Santos Coelho
Industrial and Systems Engineering Graduate Program(PPGEPS) Pontifcal Chatolic University of Paraná(PUPCR), Curitiba PR, Brazil e-mail: lscoelho2009@gmai.com

the end of the decade, yet such monumental achievement remains elusive. One of the developments in its roadmap is better diagnosis. Although there are many techniques available such as Light Microscopy, Immunochromaticgraphic Rapid Diagnostic Tests, Polymerase Chain Reaction (PCR) and Serologic Diagnosis [1], most of them are either not sensible enough or financially prohibitive. Among them, Light Microscopy has been the golden standard for a long while. However, this method requires a trained microscopist, which manually classifies and counts the Red Blood Cells (RBCs), which is very labor intensive, which makes the procedure expensive and slow. Since malaria occurs in low-income regions, in endemic bursts, effective diagnosis requires a tool that is fast and cheap. Thus, automatizing of the process has become a very desirable goal. Achieving such would make the golden standard of test cheap and readily available. With the rise of machine learning, deep learning and other computer vision tools, this has become a possibility.

1.1 Detection using Automated Methods

This process of cell classification is a natural match to automation techniques using machine learning and other image processing strategies. Many medical applications have already been demonstrated, such as Breast [2] and Lung [3] Cancer. There are complications, of course.The collection of data for such applications is troublesome and labor intensive. And since it is a medical application, there are rigorous requisites to accomplish, such as high accuracy and sensibility performance measures. Furthermore, since Malaria afflicts mostly low-income regions, there is the concern of the cost and deployment, which has to be cheap, fast and scalable. Thus, although many works have already been conducted [4–8], there are still significant improvements to be made before a mature solution is obtained.

1.1.1 Convolutional Neural Networks and Computational Complexity

Of the many automatic classification implementations, Convolutional Neural Networks (CNNs) are ubiquitous, being suitable for image classification task, having very high accuracy, sensitivity and F1-scores. However, this kind of artificial neural network usually consumes significant amounts of computational power [8], bringing into question if it is a viable solution on field. Thus, this work seeks to find CNN topologies that are highly accurate, while maintaining small computational consumption. Since training a neural network is computationally intensive, this work will take advantage of an metaheuristics-based optimization approach, using Harmony Search to comb through a search space within a reasonable runtime.

1.2 Metaheuristics Search

Metaheuristics have emerged as a solution for problems with massive search spaces, which have appeared in pretty much every area of knowledge [9,10]. Major advances have occurred in the last twenty years and the field keeps growing every day. These techniques allows for analysis of large search spaces with a fairly decent computation cost, but also yield reasonable convergence and good approximate solutions. Since many real world applications are NP-hard(non-deterministic polynomial-time hard), such quality approximate solutions are very valuable. Many metaheuristics have come forth in the last few years, such as genetic algorithms, particle swarm optimization, differential Evolution and Harmony Search [10]. Metaheuristic and neural networks have many combined applications works [11–20]. The search for neural network architectures is one of such applications.

1.2.1 Harmony Search

Harmony Search(HS) [21] is a algorithm based on musical compositions and improvisation, and has been used in several applications [22]. One of its advantages is the use of sthocastic process and non-derivative information, rendering the process computationally fast. The pseudo-code of the Harmony Search Algorithm can be seen below in Algorithm 1.

Algorithm 1 HS Pseudo-code (for discrete search space)

```
Define objective function f(x), x = (x1, x2, ..., xd)^T
Define harmony memory accepting rate (HMCR)
Define pitch adjusting rate (PAR)
Define Harmony Memory Size (HMS)
Define Maximum Pitch Adjustement Index (MPAI)
Generate Harmony Memory with HMS random harmonies
while t <max number of iterations do
    while i ≤number of variables do
        if (rand<HMCR) then
            Choose a value from HM for the variable i
            if (rand<PAR) then
                Adjust the value moving up to MPAI indexes from the current value.
            end if
        else
            Choose a random value
        end if
        Generate Harmony Memory with random harmonies
    end while
    Accept the new harmony (solution) if better
end while
Find the current best solution
```

where MPAI is the number of indexes the pitch might jump between points in a discrete the search space.

1.3 Metrics

Separating infected and non-infected RBCs is a classification task. For this kind of medical applications, there are four main measurements: Accuracy, F1-score, Precision and Recall [23, 24]. Receiver operating characteristic-Area Under Curve (ROC-AUC) is also used. Of those, F1-score is by far the most used, since encompasses both precision and Recall. The metrics can be calculated thusly:

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{1}$$

where TP is the number of cells identified as parasitized and actually parasitized, FP is the number of cells identified as parasitized, and actually non-parasitized, and FN is the number of cells identified as non-parasitized and actually parasitized. For this work, a threshold of 0.5 was chosen to calculate all F1-scores, once the models constructed give probabilities as outputs, not a binary classification.

1.3.1 F1-score considering computional cost (F1CCC)

While the F1-score is great at measuring how good any given system is at distinguishing between the two classes, it does not take into account the computational cost of the model. Since computational cost is of high importance to the solution of this problem, this work suggests a metric that takes that into account. Considering that computational complexity is often not directly measured in the development of machine learning algorithms(and even when they are, it is done is non-standardized forms, since it's heavily reliant on implementation), this work suggests that the number of parameters is used as a proxy of the actual complexity of the model. It can be defined as:

$$F1CCC = \frac{F1}{n_{params}} \tag{2}$$

where $F1$ is the F1-score and n_{params} is the number of parameters in the model.

There are problems with such metric of course. By using such metric, the risk of the search algorithm trying to sacrifice F1-score in trade of reducing the number of parameters. This is worrisome, since the magnitude of the F1-score (and its range) is significantly smaller than that of the number of parameters. To try and mitigate this, a function that reduces the significance of the number of parameters can be applied. Doing so yields the following metric:

$$F1CCC_{log} = \frac{F1}{\log_{10}(n_{params})} \tag{3}$$

where $F1$ is the F1-score and n_{params} is the number of parameters in the model. $F1CCC_{log}$ will serve as the optimization metric, and will be calculated for each iteration of the HS.

1.3.2 Performance Indicator

Since Harmony Search is stochastic, there is the concern of statistical relevancy of our results, thus, the Harmony Search has to be run multiple times, and even then, a statistical significance analysis has to be considered. For the tests in the work, a Mann-Whitney U test is performed, to verify if there is a statistical difference between the optimizations.

2 Methodology

To be able to use HS on the problem, three things have to be accomplished: The topological problem has to be describe in a finite and fixed number of variables; A objective function has to be defined and has to be able to be computed in function of the variables defined [9].

Describing a CNNs in terms of metaheuristic parameters has been done before [12]. The variables can be defined as an array y:

$$y = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 & y_5 \end{bmatrix} \tag{4}$$

where y_1 is the number of layer, y_2 is the number of starting kernels, y_3 is the size of the kernel, y_4 is the number of dense layers and y_5 is the number of nodes.

A generator code can be written to assemble a network topology with the characteristics given by y and passing that topology for training and evaluation. Encapsulating both the generation of the network as well as training and validation into a function which yields our defined metric as output will give us our optimization function.

The bounds of our search space is mostly defined by the code implementation of the network assembly, which in our case uses a Python package for neural networks, Keras. The search space for each variable is describe in the Table 1.

Once both the optimization function and the search space have been defined, a HS implementation has to be selected. In this work, the library pyHarmonySeach [28] was used.

The dataset [27] contains a total of 27,558 cell images with equal instances of parasitized and uninfected cells. A 80-20 split was made for train-test. To calculate our metrics, the topology is assembled for the parameters in y and then used in the train-test. The test dataset is then used to calculate the optimization metric (either $F1CCC$ or $F1CCC_{log}$). This feeds the harmony search algorithm as the performance

Table 1 Search Space of the Design Variables

Variables	Parameter*	Search Space
y_1	number of layers	1, 2, 3, 4, 5
y_2	number of starting kernels	2, 4, 8, 16, 32, 64
y_3	size of kernel	1, 2, 3, 4, 5
y_4	number of dense layers	1, 2, 3, 4, 5
y_5	number of nodes	16, 32, 64, 128, 256

∗ Is important to remark that there are many other hyperparameters on a Neural Network, but this work limits itself only to try to solve the topological search problem. Future works might try adding other hyperparameters as part of the search space.

metric. The parameters used in the Harmony Search were: $HMS = 10; HMCR = 0.75; PAR = 05; MPAI = 1$; HMS and $MPAI$ were mainly chosen due to the relatively small search space, while MCR and PAR were chosen as non-committal values, since tuning for these parameters would be computationally expensive. The code is developed in python and can be found here. The deployment was made on a collab notebook using Python 3 Google Compute Engine backend. A cache was implemented to return the same results for multiple runs of the same topology in a single run, such that search space remained constant during a single run. Otherwise for the same y, two different results would occur, and the search would converge poorly. It was discussed to use a average of many trains on the same topology for calculation of the F1CCC, but it incurred in significant computational overhead and the idea was dropped, but that is a point of concern for future works.

3 Results and Discussion

Five independent runs of the harmony search were calculated, for both the $F1CCC$ and $F1CCC_{log}$, each with 50 iterations ($N_{improsivations}$). Table 2 has the summary of the best results of each run.

It can be seen that while the F1-scores of both optimization metrics are very similar, they clearly diverge from the random samples. A Mann-Whitney U Test can be performed to verify this, comparing the random samples with the best results of the five runs(for each objective function separately). Doing so shows that both objective functions, p≈0.006, for every metric (F1-Score, $F1CCC$ and $F1CCC_{log}$). Meaning that is very likely that the results are indeed different. Meanwhile perform the same test between the best results of the two objective functions and p≥0.26, which implies in no statistical difference between the outcomes, implying that both metrics achieve the same results. Therefore, it can be concluded that the optimization searches in fact work, increasing the desired metric while maintaining a reasonable F1-score, however, that there is no reason to opt for one objective function ($F1CCC$) over another ($F1CCC_{log}$).

Table 2 Summary of the Classification Results

Opt. Metric*	Obs. Metric*	Mean	Best	Worst	Median
$F1CCC$	$F1$	0.9559	0.9592	0.9508	0.9570
$F1CCC_{log}$	$F1$	0.9546	0.9585	0.9512	0.9556
Random	$F1$	0.5714	0.9406	0.3290	0.3376
$F1CCC$	$F1CCC$	4.8e-05	8.6e-05	2.8650e-05	3.2320e-05
$F1CCC_{log}$	$F1CCC$	4.0e-05	7.5e-05	8.9900e-06	3.1550e-05
Random	$F1CCC$	1.0e-06	3.0e-06	1.1830e-07	6.2520e-07
$F1CCC$	$F1CCClog$	0.2207	0.2369	0.2120	0.2140
$F1CCC_{log}$	$F1CCC_{log}$	0.2139	0.2319	0.1906	0.2132
Random	$F1CCC_{log}$	0.0972	0.1730	5.2290e-02	6.4260e-02

∗ Obs. Metric being which values are we observing in the cells of the table, while Opt. Metric is the metric by which the search algorithm tried to optmize.

Sadly, it was posteriorly noted that the library used, PyHarmonySearch had a poor historic retention implementation not allowing us to plot a convergence curve, and due to computational and time constraints, another series of runs was not made. In Figure 1 the ROC-AUC can be seen for the best model for each of the strategies. The AUC is import because it allows us to visualize how can we change the True Positive Rate in exchange of False Positive Rate, which is very valuable to know when dealing with medical diagnosis. Larger AUC implies a better model. Therefore, from the AUC perspective, the models are again, highly performant.

The results of this models can be compared with other state of the art models, and can be seen in Table 3.

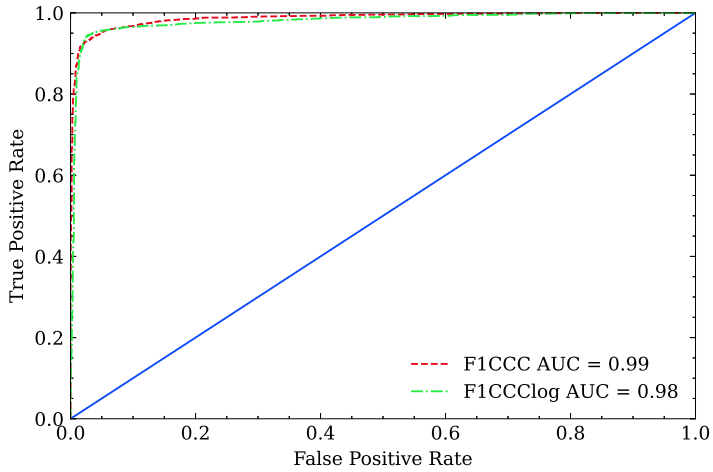

Fig. 1 ROC-AUC curve for the best model for each of the strategies

Table 3 Optimization Function Search Space

Ref	F1-Score	$F1CCC$	$F1CCC_{log}$	Nparams	Size (KB)
[4]	0.97	2.37E-6	54.43E-2	409146	-
[7]	0.99	-	-		73.70
[8]	0.98	-	-	-	-
Best F1CCC	0.95	19.4E-5	0.2584	4909	246
Best F1CCClog	0.95	7.5E-5	0.2319	12637	215

∗ Its important to note, this work did not perform a systematic review of the literature, and thus, the result comparison should be taken with a grain of salt.
+ Other important thing to take notice is the absence of the number of parameters in other works. It becomes complicated to compare outcomes when one of the results used 400 thousand parameters and another uses 10 million. Future works should strive to always disclose the number of parameters of the model, since it is highly relevant for the application.

From this table we can conclude the $F1CCC$ achieved remarkable results, obtaining a model a hundred times smaller than the one reported, while maintaining reasonable F1-scores. Meanwhile $F1CCC_{log}$ seems to have provided worse reductions of the N_{params}, nevertheless producing a 32-fold reduction. This worse performance is expected, since by applying a logarithm to the N_{params}, such factor has its impact dampened in the objective function. However it seems that the initial concern of the objective function prioritizing reducing the number of parameters instead of accuracy was misguided, and for both objective functions, F1-score remained relatively constant.

4 Conclusion and Future Works

This worked explored the utilization of a HS Algorithm in order to obtain a CNN topology for Malaria Parasite Detection. To do so it suggested function to be optimized that took parameters related to the CNN topology and assessed the resulting network by using two suggested metrics: $F1CCC$ and $F1CCC_{log}$, yielding remarkable results, reducing the number of parameters hundred-fold, while losing only a couple percentage points, definetely proving the potention of using HS to find better topologies in the CNN for malaria space. There is much to be done regarding future works,however. Better optimization functions may be suggested. Using a larger search space or a search space with more variables might be attempted or even other topology structures. Different HS hyperparameters or a auto-tuning approach can also be tested. At last, other search algorithms may be attempted, such as particle swarm optimization or differential evolution.

References

1. Bronzan, Rachel N., Meredith L. McMorrow, and S. Patrick Kachur. "Diagnosis of malaria." Molecular diagnosis therapy 12.5 (2008): 299-306.
2. Araújo, Teresa, et al. "Classification of breast cancer histology images using convolutional neural networks." PloS one 12.6 (2017): e0177544.
3. Alakwaa, Wafaa, Mohammad Nassef, and Amr Badr. "Lung cancer detection and classification with 3D convolutional neural network (3D-CNN)." Lung Cancer 8.8 (2017): 409.
4. Pattanaik, P. A., et al. "Malaria detection using deep residual networks with mobile microscopy." Journal of King Saud University-Computer and Information Sciences (2020).
5. Shekar, Gautham, S. Revathy, and Ediga Karthick Goud. "Malaria Detection using Deep Learning." 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184). IEEE, 2020.
6. Manescu, Petru, et al. "Expert-level automated malaria diagnosis on routine blood films with deep neural networks." American journal of hematology 95.8 (2020): 883-891.
7. Masud, Mehedi, et al. "Leveraging deep learning techniques for malaria parasite detection using mobile application." Wireless Communications and Mobile Computing 2020 (2020).
8. Fuhad, K. M., et al. "Deep learning based automatic malaria parasite detection from blood smear and its smartphone based application." Diagnostics 10.5 (2020): 329.
9. A Tutorial On the design, experimentation and application of metaheuristic algorithms to real-World optimization problems
10. Dokeroglu, Tansel, et al. "A survey on new generation metaheuristic algorithms." Computers Industrial Engineering 137 (2019): 106040.
11. Talbi, El-Ghazali. "Machine learning into metaheuristics: A survey and taxonomy." ACM Computing Surveys (CSUR) 54.6 (2021): 1-32.
12. Lee, Woo-Young, Seung-Min Park, and Kwee-Bo Sim. "Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm." Optik 172 (2018): 359-367.
13. Ayumi, Vina, et al. "Optimization of convolutional neural network using microcanonical annealing algorithm." 2016 International Conference on Advanced Computer Science and Information Systems (ICACSIS). IEEE, 2016.
14. de Souza, Rodrigo Clemente Thom, et al. "Binary coyote optimization algorithm for feature selection." Pattern Recognition 107 (2020): 107470.
15. Xue, Bing, et al. "A survey on evolutionary computation approaches to feature selection." IEEE Transactions on Evolutionary Computation 20.4 (2015): 606-626.
16. Optimal hyperparameter tuning of convolutional neural networks based on the parameter-setting-free harmony search algorithm
17. Kim, Seong-Hoon, Zong Woo Geem, and Gi-Tae Han. "Hyperparameter optimization method based on harmony search algorithm to improve performance of 1D CNN human respiration pattern recognition system." Sensors 20.13 (2020): 3697.
18. Awad, Noor, Neeratyoy Mallik, and Frank Hutter. "Differential evolution for neural architecture search." arXiv preprint arXiv:2012.06400 (2020).
19. Wei, Chen, et al. "Npenas: Neural predictor guided evolution for neural architecture search." arXiv preprint arXiv:2003.12857 (2020).
20. Wei, Chen, et al. "Self-supervised representation learning for evolutionary neural architecture search." IEEE Computational Intelligence Magazine 16.3 (2021): 33-49.
21. Yang, Xin-She. "Harmony search as a metaheuristic algorithm." Music-inspired harmony search algorithm. Springer, Berlin, Heidelberg, 2009. 1-14.
22. Nasir, Mohammad, et al. "Harmony Search Algorithm and Fuzzy Logic Theory: An Extensive Review from Theory to Applications." Mathematics 9.21 (2021): 2665.
23. Goutte, Cyril, and Eric Gaussier. "A probabilistic interpretation of precision, recall and F-score, with implication for evaluation." European conference on information retrieval. Springer, Berlin, Heidelberg, 2005.

24. Handelman, Guy S., et al. "Peering into the black box of artificial intelligence: evaluation metrics of machine learning methods." American Journal of Roentgenology 212.1 (2019): 38-43.
25. Carrasco, Jacinto, et al. "Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review." Swarm and Evolutionary Computation 54 (2020): 100665.
26. Derrac, Joaquín, et al. "Analyzing convergence performance of evolutionary algorithms: A statistical approach." Information Sciences 289 (2014): 41-58.
27. Rajaraman, Sivaramakrishnan, et al. "Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images." PeerJ 6 (2018): e4568.
28. https://github.com/gfairchild/pyHarmonySearch