Multi-objective Optimisation in Machine Learning

700013809

ABSTRACT

Hyperparameter optimisation is an essential process in the creation of high performing machine learning models, many of which consider multiple performance metrics to determine the quality of a solution. Therefore, these models require the use of multi-objective optimisation strategies in the exploration of different hyperparameter combinations. This research paper outlines the methods used within existing literature and details the implementation of two such methods in optimising a multilayer perceptron model designed to classify benign and malignant breast cancer samples. The results for each method are analysed and compared, alongside a base single objective optimisation method, with future works based on the limitations of the implemented methods discussed.

1 INTRODUCTION

The careful selection of hyperparameter values in machine learning is imperative to achieving a high-performing model, as modifications to the hyperparameters influence the model's topology, controlling the flow of data through the model and ultimately altering the model's behaviour. Therefore, many methods have been developed in an effort to optimise the hyperparameters of various machine learning models, often utilising a single metric/objective to assess the model's performance across a wide range of hyperparameter values. However, relying on a single performance metric can be problematic as it may oversimplify the evaluation process, potentially resulting in misleading conclusions about the model's effectiveness. For example, using accuracy alone (ratio of correct predictions to the total predictions) may lead to erroneous conclusions as it does not distinguish between the proportion of correctly classified samples for each output case. This is further emphasised in imbalanced datasets in which a model may achieve high accuracy by simply predicting the majority class, even if it performs poorly on the minority class [1].

Adopting a multi-objective approach in the optimisation of machine learning hyperparameters helps overcome the issues associated with using a single performance metric by providing a more comprehensive and balanced optimisation strategy. This methodology utilises multiple performance metrics and optimises the hyperparameters to achieve a model that provides a better trade-off between competing metrics. Therefore, ensuring a more robust and generalised model capable of performing well across a wider range of scenarios and requirements.

This research experiment focuses on the use of multi-objective optimisation in a multi-layer perceptron (MLP) network designed to classify between benign and malignant breast cancer samples. The dataset was chosen due to the high implications of false positives and false negatives. False positives lead to the incorrect diagnosis of cancer, unnecessarily scaring patients, while false negatives result in delayed treatment for those with cancer as the model fails to identify the malignant tumour. Therefore, it is crucial to develop a model that is optimised using multiple objectives to create a system that achieves a high level of accuracy whilst also maintaining a low number of false positives and false negatives. The methods used to achieve such a system are outlined in section 3, with the results of various experiments altering the optimisation methods discussed in the subsequent sections.

2 BACKGROUND CONTEXT

In recent years, machine learning has seen a substantial surge in popularity, finding applications across numerous domains and professional settings often requiring the optimisation of multiple competing objectives. Therefore, many methods have been developed in an effort to quickly and efficiently find the optimal hyperparameters to create a system that achieves high levels of performance across each objective. Three such approaches are outlined in the subsections below.

2.1 Naive Approaches

Naive approaches are the most straightforward hyperparameter optimization methods, with grid search and random search standing out as two of the most widely used techniques [1]:

- Grid search - systematically tests each possible combination of hyperparameters in a predetermined search space and stores the results.
- Random search - Randomly selects a predetermined number of samples from the hyperparameter search space and stores their results.

As grid search tests all combinations, there is a high chance of finding the optimal set of solutions. However, as the number of hyperparameters and the range of values tested increases, the number of combinations grows exponentially leading to a computationally expensive and time-consuming process. Therefore, the search space is often limited, which may result in the optimal values being omitted from the search [2]. Random search improves this efficiency as it does not require testing every combination, allowing for a larger range of values to be tested. However, its effectiveness is highly dependent on the number of iterations, as too few may fail to find the optimal solution and too many becomes too computationally expensive [1].

After running either of these methods, the results are analysed to find the Pareto set of solutions that provide the best performance across all objectives. This can be achieved through using dominance relations, finding the solutions that are not dominated in the objective space that represent the best hyperparameters for the machine learning model. Domain knowledge and objective preferences may then be used to select a single solution from the Pareto set.

2.2 Evolutionary Algorithms

Multi-objective evolutionary algorithms are commonly utilised to approximate the Pareto front and identify the optimal machine learning model's hyperparameters. These methods begin by initialising a population of solutions and evaluating their performance. Subsequently, crossover and mutation functions are used to combine features of the best solutions and make small random changes, facilitating the exploration of the solution space. The generated solutions are evaluated and inserted into the population and the worse performing solutions in the new population are removed. This process allows solutions to improve iteratively between subsequent populations with the aim of eventually finding the optimal hyperparameter solution [3]. This general process remains consistent among multi-objective evolutionary algorithms, with the selection process of the best solutions broadly categorised into three types [1]:

- Pareto dominance based - First assigns a rank to the solutions based on their dominance relations by separating them into a series of non-dominated fronts. The solutions within each rank are then ordered based on a secondary measure (e.g. crowding distance).
- Indicator based - Each solution's contribution to a metric (e.g., hypervolume) is calculated and sorted, where a higher contribution indicates a better solution.
- Decomposition Based - Splits the multi-objective problem into a number of single objective subproblems to create a scalarising function to rank solutions.

The choice of the selection function depends on each use case, with Pareto dominance based methods often performing poorly with higher numbers of objectives (>3) due to an increase in the number of non-comparable solutions. Indicator and decomposition based methods improve on this limitation but still have issues with computational efficiency with many objectives and sensitivity to the shape of the Pareto front respectively [4]. Overall, multi-objective evolutionary algorithms have demonstrated high levels of performance in hyperparameter optimisation due to their ability to improve the solution quality between iterations and search a wide range of values without testing every combination. However, multi-objective evolutionary algorithms tend to have slow convergence, requiring a large number of evaluations that may not be viable for computationally expensive machine learning problems [1].

2.3 Bayesian Optimisation

Bayesian optimisation is an iterative algorithm designed to find small improvements on solutions between each iteration. The key idea behind this method is to create a (surrogate) model that maps the hyperparameters to the problem's objective functions, that is subsequently used to suggest new hyperparameter configurations based on the previous performance. In the case of multi-objective optimisation, there are two main methods to apply Bayesian optimisation [1, 5]:

- Mono-surrogate: Create a new objective by scalarising the objective functions into a single value, using this new objective to create the surrogate model. This approach is simple to calculate but often struggles to incorporate conflicting objectives.
- Multi-surrogate: Create a separate surrogate model for each objective and use a multi-objective optimiser to generate the next hyperparameter combination. Using multiple models aims to create a more robust and versatile optimisation process able to cope with conflicting objectives. However, the calculation and updating of multiple surrogate models results in a more computationally expensive system [5].

Bayesian optimisation has achieved notable success within multi-objective optimisation problems due to its ability to learn the relationships between the system's hyperparameters and objectives to iteratively improve solutions. This is especially evident in the optimisation of computationally expensive machine learning models, as the comparatively low number of hyperparameter evaluations and absence of a population of solutions results in a high-performing and efficient optimisation strategy [6].

3 METHODOLOGY

As previously discussed, the goal of this research experiment is to create a multi-objective optimiser to optimise the hyperparameters of a multi-layer perceptron classifier for the identification of benign and malignant breast cancer samples. The implementation of such a system is split into two sections: the development of the classifier and the creation of the optimisation methods.

3.1 Classification System

Before creating the multi-layer perceptron model, it is necessary to process the dataset and its features, transforming them into a format that can be more easily interpreted by the model. The specific steps employed in this research experiment to accomplish this are outlined below:

- Feature normalisation - ensures the features are of similar scales to enable the use of larger learning rates without focusing too much on the features with larger initial values.
- Feature selection - as the dataset has 30 features, a random forest classifier is used to sort the features by importance and remove the features that have minimal contribution to the end classification. Reducing the model's complexity and allowing for a wider range of hyperparameter values to be tested whilst maintaining performance.
- Stratified sampling - split the data into training a testing sets, maintaining the proportion of benign and malignant samples from the original dataset.

The multi-layer perceptron classification model is implemented using scikit-learn's MLPClassifier function and is trained with 5-fold cross-validation for each solution. This approach helps detect overfitting and ensures that the training performance accurately represents the model's capabilities on unseen data.

3.2 Optimisation Methods Implemented

Three different hyperparameter optimisation methods were implemented in this experiment, with the results of each method presented in section 4. Each of these methods and their implementations are outlined below:

Single Objective Gridsearch - this method employs a simple grid search (explained in Section 2.1) to test all possible combinations of the following MLP classification hyperparameters:

- hidden_layer_sizes: alters the number and size of the hidden layers, 9 values tested from **(5,25,5) to (15,100,15)**
- activation: the activation function used, values tested **tanh, relu & identity**
- alpha: strength of L2 regularisation, values tested **0.0001 & 0.05**
- batch_size: number of samples used before updating the model's internal parameters, values tested **250 & 300**

These values result in 108 possible hyperparameter combinations, in which the accuracy alone is used to determine the best solution within the search space. However, it is crucial to note that, given the computationally expensive nature of MLP model training, the search space was limited to ensure a feasible runtime

Multi-objective Pareto Front Gridsearch - The initial steps in this method remain consistent with the single-objective grid search, testing the same 108 hyperparameter combinations on the MLP classification system. However, in this method the true positive rate (TPR) and false positive rate (FPR) are calculated and stored along with the accuracy of each combination. The TPR and FPR were chosen due to the high implications that result from having a large proportion of false positive and false negative results (explained in Section 1). The stored results are then analysed to find the solutions that minimise the FPR and maximise the accuracy and TPR (or minimise -1*accuracy & -1*TRP). To achieve this, non-dominated sorting is used to find the Pareto front of solutions that are not dominated by any other solution in the objective space, representing the hyperparameter configurations with the best performance across all metrics. In this experiment, a solution from the Pareto front is selected by calculating the equally weighted sum of each objective. However, this selection criterion can easily be adjusted based on objective preferences and domain knowledge.

Multi-objective Bayesian Optimisation - This final method utilises Bayesian optimisation (explained in Section 2.3) in an attempt to find the best solution within the following search space:

- hidden_layer_sizes: Layer 1 & 3: 5-50 with step size 5, layer 2: 10-300 with step size 10
- activation: tanh, relu & identity
- alpha: 0.0001-0.1 with step size 0.001
- batch_size: 200-300 with step size 20

This method, implemented with the hyperopt package, creates a surrogate model by minimizing a scalarising function with equal weight for each objective, used to improve solutions over each iteration. This method was chosen over other optimisation methods (e.g. evolutionary algorithms) due to the lower number of computationally expensive evaluations needed to converge on a high performing solution.

For each of the previously described optimisation methods, the solution found with the highest performance across all metrics is used to predict the unseen test set. The average accuracy, true positive rate (TPR), and false positive rate (FPR) across multiple runs are then compared for each optimizer and displayed in Section 4.

3.3 Hypothesis

The single objective optimisation approach is likely to achieve results with high levels of accuracy. However, a higher number of false positives and false negatives compared to other methods is likely, as this approach does not take into account the TPR and FPR. The multi-objective Pareto front grid search is likely to address this limitation by including the TPR and FPR in the optimiser. However, the limited search space made necessary by the computationally expensive machine learning task may result in the optimal values being omitted from the search space. Bayesian optimisation can, however, efficiently utilise a large search space whilst obtaining solutions that provide high performance across all metrics. Therefore, Bayesian optimisation is likely to provide the best results of the three methods described.

4 RESULTS AND ANALYSIS

4.1 Evaluation of Solutions Found

Using a random state of 100 for the feature selection and MLP classification model, the three optimisation methods return solutions with the following performance metrics:

	Test Acc.	TPR	FPR
Sing. Obj	91.23%	92.86%	9.72%
PF Gridsearch	92.98%	95.23%	8.33%
Bayesian Opt	96.49%	95.24%	2.78%

Additionally, in this run, cross-validation training accuracy values were identical for both the single-objective and Pareto front grid search methods, at 95.16%. When predicting the test set, both methods experienced a decrease in accuracy, with the Pareto front grid search (PFGS) method returning a slightly higher test accuracy. However, the key difference lies in the TPR and FPR, where the PFGS method shows a higher TPR and lower FPR. Therefore, in this run, PFGS outperformed the single-objective method across all metrics. Furthermore, in this run, the best solution found by the single objective method was also present in the non-dominated solutions found by the PFGS method. While this trend is often observed between runs, the two methods rarely return the same best solution due to the different metrics and selection criteria used.

At this random seed, the Bayesian optimization method (run for 60 iterations) emerges as the highest-performing model, yielding significantly higher accuracy, slightly higher TPR, and significantly lower FPR compared to the next best method (PFGS). Additionally, the hyperparameters of the solution returned from this method exhibit only small variations from the values present within the limited search space of the PFGS and single-objective methods.

4.2 Average Performance

The average performance of each parameter optimization method is assessed by calculating the mean accuracy, true positive rate, and false positive rate over 10 separate runs, adjusting the random state seed in each run. The average values for each optimisation method are shown in Figure 1 below.

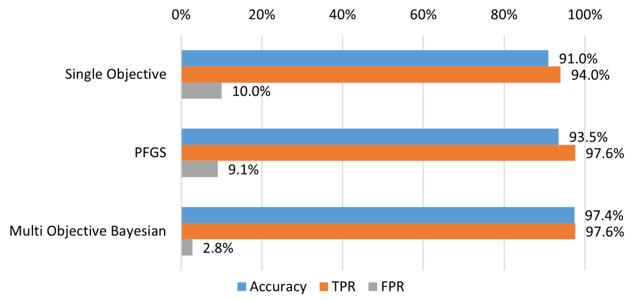

Figure 1: Average metric values for each method

The results in this figure validate the findings from the previously discussed run with a random state of 100. Bayesian optimization outperforms the other two methods in every metric, demonstrating significant improvements in accuracy and FPR whilst achieving a TPR equal to the next best performing method (PFGS). Additionally, PFGS again exhibits improved performance across all metrics compared to the single objective optimisation method, with slight improvements of 1-3% for each metric.

Additionally, it is important to note that although the multi-objective Bayesian optimisation method improves upon each performance metric, the key difference is found within the false positive rate. The single objective and PFGS methods both exhibit a relatively high FPR, with average values of 10% and 9.1% across the evaluated runs respectively. In contrast, the multi-objective Bayesian optimization method significantly reduces this value to 2.8%. Therefore, the key improvement multi-objective Bayesian optimisation method exhibits over the other methods is its ability to correctly classify the negative (benign) samples. This is further emphasised by plotting the confusion matrix, displayed for the PFGS (left) and multi-objective Bayesian optimisation (right) methods below:

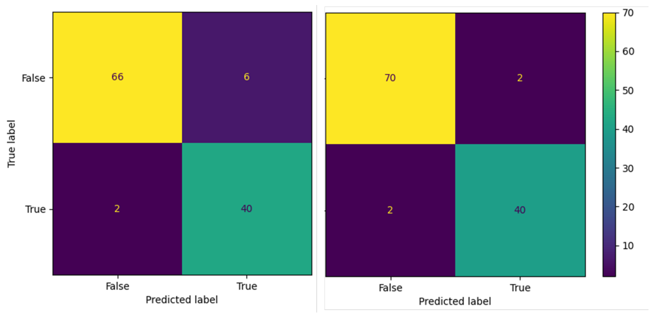

Figure 2: Confusion Matrices

As displayed, the values for true positives and false negatives in this run remain consistent, resulting in an equal true positive rate between the PFGS and multi-objective Bayesian optimisation methods. However, the number of false positives is significantly lower in the multi-objective Bayesian optimisation method, resulting in a lower false positive rate, consistent with the average results previously discussed.

The results presented within this section align with expectations, confirming the hypothesis outlined in Section 3.3. The single objective grid search method achieves relatively high levels of accuracy. However, since the optimisation relies on accuracy alone, the algorithm does not distinguish between the number of correctly classified samples for each output case. This results in a higher proportion of false positives and false negatives, leading to a higher false positive rate and lower true positive rate than the other methods used.

The multi-objective Pareto front gridsearch method (PFGS) improves upon these results by incorporating the TPR and FPR in the solution evaluation process. This enhances the machine learning model's ability to correctly distinguish between benign and malignant samples, reducing the average false positive rate from 10% to 9.1% and increasing the true positive rate from 94% to 97.6%. Additionally, this method returns a number of non-dominated solutions, each of which are statistically equivalent in their overall performance, allowing for a solution to be chosen based on objective preference and domain knowledge. However, a key limitation of this method is the exponential growth in the number of combinations when increasing the number of hyperparameters and values tested. Consequently, the search space must be limited, making it impossible to find small variations in the model's parameters that could lead to the optimal model, thereby limiting the achievable performance.

Ultimately, multi-objective Bayesian optimisation achieves the best results, reducing the average false positive rate to 2.8% and increasing the average accuracy and true positive rate to 97.4% and 97.6% respectively. This success can likely be attributed to the utilisation of multiple performance metrics and the development of a surrogate model that maps hyperparameter values to objective functions. This mapping facilitates the use of a large search space without compromising the system's efficiency. Therefore, increasing the likelihood of identifying the optimal hyperparameters, resulting in a machine learning model with reduced instances of false positives and false negatives.

This conclusion is further backed by the results displayed within various published multi-objective hyperparameter optimisation research papers. One such paper reports that the use of a mono-surrogate multi-objective Bayesian optimisation using a scalarised loss function outperforms both a random search and the NSGA-II evolutionary algorithm. In which the Bayesian optimisation solutions consistently dominate a larger region of the objective space than either of the other two methods [7]. Additionally, a second research paper concludes that multi-objective Bayesian optimisation methods provide an efficient optimisation approach that can find the optimal trade-offs between conflicting objectives [8].

5 DISCUSSION AND FUTURE WORK

This paper outlined three hyperparameter optimization methods: a single-objective grid search, a multi-objective grid search with Pareto front analysis, and a multi-objective Bayesian optimization. The results of multiple test runs were compared, indicating that Bayesian optimization consistently outperformed the other two by reducing false positives and false negatives in the classification of benign and malignant breast cancer samples. Although this method achieves high levels of performance, future works to potentially improve the efficiency and performance of the methods described in this paper are outlined below:

- Multi-surrogate Bayesian optimisation: As discussed in Section 2.3, multi-surrogate Bayesian optimization involves creating a surrogate model for each objective function. This approach aims to develop a more robust system compared to the mono-surrogate approach used in this project, in which the scalarising function may remove important information from the objective landscape [5]. Therefore, future work could entail implementing a multi-surrogate approach that is able to more accurately represent the objective landscape and generate higher performing solutions.
- Utilise the surrogate model to select a solution from the Pareto front: Other works have demonstrated the capability of using the surrogate model created during the Bayesian optimization process to make a more informed and robust selection from the Pareto front [9]. Therefore, future works could implement such a method to make a more informed decision than the simple equally weighted sum used in this project.

REFERENCES

[1] F. Karl, T. Pielok, J. Moosbauer, F. Pfisterer, S. Coors, M. Binder, L. Schneider, J. Thomas, J. Richter, M. Lang *et al.*, "Multi-objective hyperparameter optimization–an overview," *arXiv preprint arXiv:2206.07438*, 2022.

[2] H. Alibrahim and S. A. Ludwig, "Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization," in *2021 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2021, pp. 1551–1559.

[3] V. L. Vachhani, V. K. Dabhi, and H. B. Prajapati, "Survey of multi objective evolutionary algorithms," in *2015 international conference on circuits, power and computing technologies [ICCPCT-2015]*. IEEE, 2015, pp. 1–9.

[4] Y. Xu, H. Zhang, L. Huang, R. Qu, and Y. Nojima, "A pareto front grid guided multi-objective evolutionary algorithm," *Applied Soft Computing*, vol. 136, p. 110095, 2023.

[5] C. Stock-Williams, T. Chugh, A. Rahat, and W. Yu, "What makes an effective scalarising function for multi-objective bayesian optimisation?" *arXiv preprint arXiv:2104.04790*, 2021.

[6] M. Parsa, J. P. Mitchell, C. D. Schuman, R. M. Patton, T. E. Potok, and K. Roy, "Bayesian multi-objective hyperparameter optimization for accurate, fast, and efficient neural network accelerator design," *Frontiers in neuroscience*, vol. 14, p. 667, 2020.

[7] J. Knowles, "Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE transactions on evolutionary computation*, vol. 10, no. 1, pp. 50–66, 2006.

[8] X. Lin, Z. Yang, X. Zhang, and Q. Zhang, "Pareto set learning for expensive multi-objective optimization," *Advances in Neural Information Processing Systems*, vol. 35, pp. 19 231–19 247, 2022.

[9] R. Calandra, J. Peters, and M. Deisenrothy, "Pareto front modeling for sensitivity analysis in multi-objective bayesian optimization," in *NIPS Workshop on Bayesian Optimization*, vol. 5. Citeseer, 2014.