Automated method design for cancer image classification by Differential Evolution and Ensembling

Anonymized Authors

Anonymized Affiliations
email@anonymized.com

Abstract. Developing deep learning models for cancer image classification requires many method design choices, such as in data preprocessing, model architecture, hyperparameters and training procedures. Typically, these are manually tuned, a process that is time-consuming, expert-dependent, and often irreproducible. To address these challenges, we propose an automated machine learning (AutoML) framework that optimizes model design without human intervention. To ensure a comprehensive exploration of diverse architectures and hyperparameter configurations, we define a search space based on state-of-the-art literature in cancer imaging. Our framework employs Differential Evolution and Hyperband (DEHB), which integrates evolutionary search algorithms to balance search space exploration and exploitation, combined with adaptive resource allocation to mitigate the high computational cost of training multiple models. To enhance model robustness and reduce overfitting in data-limited scenarios, we incorporate ensembling. We validate our approach on four public cancer classification datasets encompassing 750 patients with either MRI or CT. The proposed framework demonstrates higher performance when compared to a DenseNet-121 baseline. While exploring multiple configurations, our approach reduces training time by a factor of two to five compared to the baseline. By automating model design and improving generalization across datasets, the framework has substantial potential for broad applications across cancer imaging, thereby streamlining deep learning model development. Our code is available on Github: Project Repository

Keywords: AutoML · evolutionary algorithms · hyperparameter optimization · computer-aided diagnosis

1 Introduction

Deep learning has substantially advanced cancer image analysis by enabling precise and early diagnosis through AI-driven models [1]. However, developing these models remains a labor-intensive process that requires numerous decisions regarding data preprocessing, model architecture, hyperparameters, and model training. Commonly the majority of these choices are made manually through a

heuristic trial-and-error process, which is time-consuming, prone to overfitting [2], limits reproducibility [3,4], may lead to suboptimal solutions [3] and can be resource-intensive, making it inefficient [5,6].

Among cancer imaging tasks, classification poses unique challenges, as it often relies on a limited number of image-level labels. Resulting in small, heterogeneous datasets that make model development more sensitive to hyperparameter choices and prone to overfitting, where models may learn spurious correlations instead of generalizable patterns [7]. To address these challenges, we propose an automated method design framework based on Automated Machine Learning (AutoML) [3] to streamline and optimize deep learning model development for cancer image classification. Our framework automates key design decisions, including data preprocessing, model selection, hyperparameter tuning, and model training, thereby eliminating manual intervention. We leverage Differential Evolution and Hyperband (DEHB) [8], a state-of-the-art AutoML optimization algorithm, to efficiently explore large hyperparameter spaces and improve both convergence speed and model performance. We hypothesize that integrating DEHB into the model development pipeline will substantially accelerate training compared to manual tuning. Additionally, we incorporate ensembling strategies to regularize DEHB optimization and improve overall performance.

2 Methods

To automate model design, given a training dataset D_{train} and a validation dataset D_{val}, we define an optimization function with the objective of identifying the optimal hyperparameter configuration x^* that minimizes the average validation loss $\mathcal{L}$ over K_{training} iterations (e.g., cross-validation). We formulate the optimization of any method design choice, including data preprocessing, model architecture, and model training, as a combined method selection and hyperparameter optimization problem. Formally, we express this as follows:

$$x^* = \arg\min_{x \in X} \frac{1}{K_{\text{training}}} \sum_{k=1}^{K_{\text{training}}} \mathcal{L}(x, D_{\text{train},k}, D_{\text{val},k}, b), \tag{1}$$

where x represents a hyperparameter configuration, X denotes the hyperparameter search space, and a given computational budget b (i.e., number of epochs).

Manual tuning and conventional tuning strategies such as exhaustive grid search or random search over X are computationally expensive, as these methods lack efficient resource allocation mechanisms. To address this, we propose to employ DEHB, which effectively balances the exploration of the search space with a focus on promising solutions through an evolutionary algorithm, while efficiently managing computational resources using Hyperband. By leveraging Differential Evolution (DE) [10] for guided search and Hyperband [6] for adaptive resource allocation, DEHB provides a more efficient and scalable approach to hyperparameter optimization, making it particularly well-suited for large and complex search spaces as well as resource-intensive model training.

2.1 Differential Evolution and Hyperband (DEHB)

As presented in Algorithm 1, DEHB is initialized by evaluating a population of Q candidates, i.e., set configurations of hyperparameters randomly sampled from X. Initially, each configuration is trained on $b_{\min}$, and its validation loss $\mathcal{L}(x, D_{\text{train}}, D_{\text{val}}, b_{\min})$ is measured. After initialization, Hyperband dynamically allocates computational resources by discarding poorly performing configurations in successive halving steps. Configurations are evaluated in progressive stages, where in each stage only the top $1/\eta$ fraction of candidates continues to the next stage with an increased budget b, while the rest are eliminated. Parameter η, known as the aggressiveness factor, controls how many configurations are discarded. The number of remaining candidates at stage s is therefore $Q \cdot (1/\eta)^s$, making the reduction schedule deterministic. The budget b increases at each stage according to the Hyperband schedule, typically growing multiplicatively with η. This process iterates until a predefined $b_{\max}$ is reached or only one configuration remains. Once Hyperband has selected the top-performing candidates for further evaluation, DEHB extends this process by continuing the search with evolutionary steps. The general procedure stops once all P configurations have been evaluated and the x^* is selected based on the validation loss.

Algorithm 1 Differential Evolution and Hyperband (DEHB)

Require: Search space $\mathcal{X}$, budgets $b_{\min}$, $b_{\max}$, reduction factor η, max evaluations P
Randomly sample Q configurations from $\mathcal{X}$ into $P_{b_{\min}}$
Evaluate all configurations in $P_{b_{\min}}$ at budget $b_{\min}$
while total function evaluations $< P$ **do**
  for each budget b in $\{b_{\min}, \eta b_{\min}, \eta^2 b_{\min}, \ldots, b_{\max}\}$ **do**
    Select top $1/\eta$ fraction of configurations from previous budget as parent pool
    for each configuration x_i in the parent pool **do**
      Generate mutant vector v_i
      Generate trial vector u_i via crossover between x_i and v_i
      Evaluate u_i on budget b
      Replace x_i in the population if u_i performs better
    end for
  end for
end while
return x^*: best configuration across all budgets

Mutation Each configuration x_i represents a vector of chosen hyperparameters. New candidate configurations, or mutant vectors v_i, are generated through a mutation process that introduces controlled variations based on existing ones. Specifically, mutation is performed as follows:

$$v_i = x_r + F \cdot (x_a - x_b),$$

where x_r, x_a, x_b are randomly selected hyperparameter configurations from the population, and F is a scaling factor controlling the magnitude of change within the range (0,1].

Crossover After mutation, a new candidate vector u_i is generated through crossover, where each parameter is inherited from the original vector x_i or the mutant vector v_i. Each configuration vector (e.g., x_i) consists of N hyperparameters, denoted as x_i^j, v_i^j, and u_i^j, where $j \in \{1, 2, \ldots, N\}$ indexes the individual hyperparameters. Crossover is performed independently for each hyperparameter j, selecting between v_i^j and x_i^j according to the following rule:

$$u_i^j = \begin{cases} v_i^j, & \text{if } (z \sim \mathcal{U}(0,1)) \leq Cr \text{ or } (j = j_{\text{rand}}) \\ x_i^j, & \text{otherwise} \end{cases}$$

where Cr is the crossover probability, determining the likelihood of inheriting parameters from v_i. The index j_{rand}, randomly selected from $\{1, 2, \ldots, N\}$, ensures that at least one parameter comes from v_i, to prevent $u_i = x_i$. This process allows crossover to integrate information from the mutant vector while preserving useful characteristics from the original configuration.

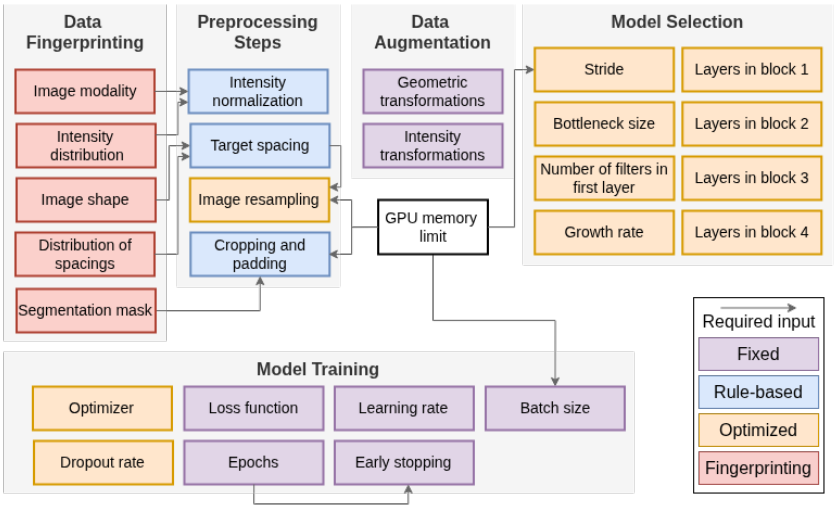
Selection The selection process then compares the performance of u_i against its corresponding parent configuration x_i using the optimization function in Equation 1. If u_i achieves a lower validation loss than x_i, it replaces x_i in the population for the next iteration. Otherwise, x_i is retained.

2.2 Search space

The performance of an AutoML framework highly depends on the search space, i.e., the potential hyperparameter configurations. Since non-architectural design choices such as data preprocessing, augmentation, model selection, and model training can be equally important as architectural ones [11,12], we construct a comprehensive search space covering all these aspects, including discrete and categorical hyperparameters, which motivates our choice of DEHB as the optimization strategy. To reduce computational burden, we set a GPU memory constraint to filter out infeasible configurations and maintain efficient training times. The complete search space is illustrated in Figure 1.

Inspired by nnU-Net [11], we introduce data fingerprinting to collect key information about the input dataset, including image modalities, intensity distribution, image shape, and spacing, which are used to determine various rule-based hyperparameters.

Preprocessing includes intensity normalization and resampling. To account for differences in resolution due to scanner variations, we included different target spacings. For cropping and padding, we constrained the image sizes to a maximum of 512 voxels in all three dimensions. As our datasets contain the tumor segmentation masks, we applied tumor-centered cropping to focus on the region of interest while reducing unnecessary background.



Design choice	Configuration
Intensity normalization	MRI: per-image z-score normalization. CT: global percentile clipping followed by z-score normalization.
Target spacing	Median, mean, fixed isotropic (1×1×1 mm), and isotropic based on the median volume.
Cropping and padding	Crop the bounding box if its dimensions exceed available memory, and pad if any dimension is below the model's minimum size.
Image resampling	Linear interpolation
Optimizer	SGD with Nesterov momentum (μ = 0.99), Adam or Adadelta
Dropout rate	{**0.0**, 0.1, 0.2, 0.3, 0.4, 0.5}
Bottleneck size	{1, 2, **4**}
Number of layers in the first layer	{16, 32, **64**}

Design choice	Configuration
Growth rates	{12, 16, 20, 26, **32**}
Layers in block 1	{4, **6**, 8, 10}
Layers in block 2	{8, **12**, 16, 20}
Layers in block 3	{16, **24**, 32, 40}
Layers in block 4	{8, 12, **16**, 20}
Stride	{1, **2**, 3}
Data augmentation	Flipping, rotations, zooming, scaling, and adjusting contrast
Loss function	Cross-entropy loss
Learning rate	ExponentialLR scheduler that decays the learning rate by a factor of 0.9
Epochs	Min: 30, Max: 50
Batch size	Batch size 1 and gradient accumulation after 5 epochs.
Early stopping	40% of the number of epochs

Fig. 1. Overview of the proposed search space for cancer image classification. Default values are shown in bold.

The backbone architecture used is DenseNet [13], which has demonstrated strong performance in cancer imaging tasks due to its efficient feature propagation and parameter efficiency, making it suitable for learning from limited and heterogeneous datasets. Search ranges for bottleneck size, growth rate, initial filter count, and block depth were defined based on default DenseNet-121 values [13] and expanded to explore broader architectural variations.

Training parameters, including the loss function [14] and learning rate [15], were fixed based on previous literature work to balance performance and overfitting. To ensure stable optimization with a batch size of one, gradient accumulation was combined with instance normalization to maintain consistent updates and training dynamics.

2.3 Ensembling

To mitigate overfitting and improve generalization, we construct ensembles by averaging predictions from multiple trained models. Given a set of M selected hyperparameter configurations $\{x_1, \ldots, x_M\}$, each of which is evaluated across K different data splits (e.g., cross-validation folds), we obtain a total of $M \cdot K$ trained models.

The ensemble prediction $\hat{y}$ for an input z is computed as

$$\hat{y} = \frac{1}{M \cdot K} \sum_{m=1}^{M} \sum_{k=1}^{K} \hat{y}_{m,k}(z),$$

where $\hat{y}_{m,k}(z)$ denotes the prediction made by the model trained with configuration x_m on fold k. This formulation allows for flexibility in ensemble design: setting $M = 1$ yields an ensemble over data splits, while increasing M incorporates configuration-level diversity.

3 Experiments and results

3.1 Dataset

For this study, four datasets from the publicly available WORC Database [16] are used, comprising 750 anonymized patients with various tumor types. Each dataset includes an MRI or CT scan, a tumor segmentation, and a tumor type label serving as the prediction target. The datasets contain 115 patients with liposarcoma or lipoma (Lipo), 203 with desmoid-type fibromatosis or extremity soft-tissue sarcomas (Desmoid), 186 with primary solid liver tumors (Liver), and 246 with gastrointestinal stromal tumors or similar intra-abdominal tumors (GIST).

3.2 Experimental Setup

To evaluate the framework, two baselines were used: (1) the WORC framework [17], previously validated on these datasets, and (2) DenseNet-121 [13] without hyperparameter tuning, trained for 100 epochs with 5-fold cross-validation using default settings.

For DEHB, the maximum and minimum fidelity levels (number of epochs) were set to $b_{\max} = 50$ and $b_{\min} = 30$, respectively. Other parameters followed default values [8]: $\eta = 2$, $F = 0.5$. Each dataset was split into 80% training and 20% test sets. All model selection and cross-validation procedures, including those for DEHB and DenseNet-121, were conducted within the training set using 5-fold cross-validation. DEHB performed $P = 10$ function evaluations per fold, selecting one best configuration per fold. These five configurations were retrained and evaluated across all five folds to obtain robust performance estimates, yielding 25 trained models per dataset. All experiments were run on 8×NVIDIA A40 48 GB GPUs using MONAI [18] for efficient medical image processing.

Model performance was primarily measured using ROC-AUC [19]. DEHB was evaluated in three ways: (1) the best configuration (lowest average validation loss across five folds) was retrained on the full 80% training set and tested on the 20% holdout; (2) a 5-Model Ensemble (5ME) was created by training the top configuration across all five folds ($M = 1$, $K = 5$); and (3) a 25-Model Ensemble (25ME), where the top five configurations were retrained across five folds ($M = 5$, $K = 5$), and their predictions averaged.

Computation time Wall-clock time was measured for each dataset to assess the efficiency of our framework, using $P = 50$ configurations. For the DenseNet-121 baseline, training was performed with default settings using a single configuration. Since training all P configurations for the baseline was computationally intensive, we estimated the cost by training once on the larger datasets (Desmoid and GIST) and averaging five runs on the smaller ones (Lipo and Liver). The total cost for $P = 50$ was extrapolated from these runs. In contrast, our framework trained all P configurations directly, enabling a fair comparison of training time and computational cost between DEHB and the baseline.

3.3 Results

Table 1 compares the performance of different methods across all four datasets. The DEHB framework alone slightly underperforms both baselines on most datasets. However, incorporating the 5ME strategy with DEHB improves performance across most datasets, with a higher performance increase observed with DEHB + 25ME in all datasets.

Computation time Our framework showed substantially lower computational cost across all datasets compared to the DenseNet-121 baseline. On Lipo, DEHB completed all $P = 50$ configurations in 95 hours versus 650 for the baseline. For Liver, it required 220 hours (vs. 900), and for Desmoid, 340 hours (vs. 1100). On GIST, the largest dataset, DEHB finished in 216 hours (9 days), while the baseline was projected to take 3600 hours (150 days). These results demonstrate a 2–5 times speedup, indicating that the framework scales efficiently and enables automated model design without compromising runtime feasibility.

Table 1. ROC-AUC for the four datasets, evaluated under three different setups: (1) WORC in 100× random-split cross-validation, (2) DenseNet-121 in 5-fold cross-validation, and (3) DEHB using a fixed 80/20 train/test split.

Model	**Lipo**	**Liver**	**Desmoid**	**GIST**
Baselines				
WORC	0.83	0.80	0.82	0.77
DenseNet-121	0.82	**0.88**	0.80	0.61
Framework				
DEHB	0.78	0.83	0.73	0.73
DEHB + 5ME	0.83	0.86	0.80	0.78
DEHB + 25ME	**0.87**	0.87	**0.83**	**0.81**

4 Discussion and conclusion

This study presents an AutoML framework for cancer image classification that combines DEHB with ensembling for automated method design. DEHB identified multiple high-performing models with similar validation performance, as observed in five-fold cross-validation. When ensembled, these models showed improved generalization, suggesting they overfit to different aspects of the training data. Ensembling helps counteract overfitting by averaging individual model biases. Moreover, DEHB substantially reduced computational time compared to the baseline by leveraging adaptive resource allocation, avoiding exhaustive training and minimizing the need for manual tuning. Further efficiency gains may be possible through techniques like reduced precision computations [20].

While our evaluation offers a practical comparison basis, the fixed 80/20 train/test split used for DEHB may limit comparability with baselines trained under different protocols. Future work will focus on aligning evaluation strategies. Additionally, although DenseNet-121 was used as the backbone, extending the framework to other architectures could offer broader insight into DEHB's generalizability.

Our results highlight the effectiveness of combining DEHB with ensembling to achieve competitive performance while significantly reducing computational cost. By efficiently exploring configurations across preprocessing, architecture, and training, the framework enables fully automated model design. This not only accelerates development but also improves reproducibility, reduces overfitting, and avoids suboptimal choices. These strengths make DEHB a compelling solution for optimizing deep learning in cancer imaging, especially in resource-constrained settings, and support more scalable, reproducible AI-driven diagnostics.

References

1. Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., Sánchez, C.I.: A survey on deep learning in medical image analysis. Medical Image Analysis **42**, 60–88 (2017)

2. Hosseini, M., Powell, M., Collins, J., Callahan-Flintoft, C., Jones, W., Bowman, H., Wyble, B.: I tried a bunch of things: The dangers of unexpected overfitting in classification of brain data. Neuroscience & Biobehavioral Reviews **119**, 456–467 (2020)
3. Hutter, F., Kotthoff, L., Vanschoren, J.: Automated Machine Learning: Methods, Systems, Challenges. Springer Nature (2019)
4. Shen, D., Wu, G., Suk, H.I.: Deep learning in medical image analysis. Annual Review of Biomedical Engineering **19**(1), 221–248 (2017)
5. Yang, L., Shami, A.: On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing, **415**, 295–316 (2020)
6. Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Hyperband: A novel bandit-based approach to hyperparameter optimization. In: International Conference on Learning Representations (ICLR) (2017)
7. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2921–2929 (2016)
8. Awad, N., Mallik, N., Hutter, F.: DEHB: Evolutionary Hyperband for Scalable, Robust and Efficient Hyperparameter Optimization. arXiv preprint arXiv:2105.09821 (2021)
9. Dietterich, T.G.: Ensemble methods in machine learning. In: International Workshop on Multiple Classifier Systems, pp. 1–15. Springer, Berlin, Heidelberg (2000)
10. Storn, R.M., Price, K., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Springer Science & Business Media (2006)
11. Isensee, F., Jaeger, P.F., Full, P.M., Vollmuth, P., Maier-Hein, K.H.: nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation. Nature Methods **18**, 203–211 (2021)
12. de Raad, K.B., et al.: The Effect of Preprocessing on Convolutional Neural Networks for Medical Image Segmentation. In: IEEE 18th International Symposium on Biomedical Imaging (ISBI), pp. 655–658. Nice, France (2021). https://doi.org/10.1109/ISBI48211.2021.9433952
13. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely Connected Convolutional Networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4700–4708 (2017)
14. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
15. Li, Z., Arora, S.: An Exponential Learning Rate Schedule for Deep Learning. arXiv preprint arXiv:1910.07454 (2019)
16. Starmans, M.P.A., Timbergen, M.J.M., Vos, M., Padmos, G.A., Grünhagen, D.J., Verhoef, C., Sleijfer, S., van Leenders, G.J.L.H., Buisman, F.E., Willemssen, F.E.J.A., Groot Koerkamp, B., Angus, L., van der Veldt, A.A.M., Rajicic, A., Odink, A.E., Renckens, M., Doukas, M., de Man, R.A., IJzermans, J.N.M., Miclea, R.L., Vermeulen, P.B., Thomeer, M.G., Visser, J.J., Niessen, W.J., Klein, S.: The WORC* database: MRI and CT scans, segmentations, and clinical labels for 930 patients from six radiomics studies. medRxiv preprint (2021). https://doi.org/10.1101/2021.08.19.21262238
17. Starmans, M.P.A., van der Voort, S.R., Phil, T., Timbergen, M.J.M., Vos, M., Padmos, G.A., Kessels, W., Hanff, D., Grünhagen, D.J., Verhoef, C., Sleijfer, S., van den Bent, M.J., Smits, M., Dwarkasing, R.S., Els, C.J., Fiduzi, F., van Leenders, G.J.L.H., Blazevic, A., Hofland, J., Brabander, T., van Gils, R.A.H., Franssen, G.J.H., Feelders, R.A., de Herder, W.W., Buisman, F.E., Willemssen, F.E.J.A., Groot Koerkamp, B., Angus, L., van der Veldt, A.A.M., Rajicic, A., Odink, A.E.,

Deen, M., Castillo T., J.M., Veenland, J.F., Schoots, I., Renckens, M., Doukas, M., de Man, R.A., IJzermans, J.N.M., Miclea, R.L., Vermeulen, P.B., Bron, E.E., Thomeer, M.G., Visser, J.J., Niessen, W.J., Klein, S.: Reproducible Radiomics through Automated Machine Learning Validated on Twelve Clinical Applications. arXiv preprint arXiv:2108.08618 (2021)

18. MONAI Consortium: MONAI: Medical Open Network for AI. Zenodo (2024). https://doi.org/10.5281/zenodo.13942962
19. Fawcett, T.: ROC Graphs: Notes and Practical Considerations for Researchers. In: Machine Learning, **31**(1), 1–38 (2006)
20. Selvan, R., Schön, J., Dam, E.B.: Operating critical machine learning models in resource-constrained regimes. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 325–335. Springer, Cham (2023)