Hyperparameters Optimization of Deep Learning Models for Unsupervised Lung Cancer Detection

Najeh Nafti
National Engineering School of Sfax
Medical Imaging and technology Lab
Sfax, Tunisia

Olfa Besbes
COSIM Lab, SUPCOM,
ISITCOM, Sousse University, Tunisia

Asma Ben Abdallah
Medical Imaging and technology Lab
ISIMM
Monastir, Tunisia

Mohammed Hedi Bedoui
Medical Imaging and technology Lab
Faculty of Medicine of Monastir
Monastir, Tunisia

***Abstract*—Lung cancer remains a significant global cause of mortality, affecting populations worldwide. Deep Learning (DL) systems show promise in early detection using clinical data to reduce mortality rates. However, these systems heavily rely on large amounts of annotated anomalous data, and heavily depends on selecting appropriate hyperparameters that define the network's structure and learning process.**

In this study, we propose an optimization scheme based on Tree Parzen Estimator (TPE) and Bayesian optimization (BO) algorithms for hyperparameter optimization in unsupervised lung cancer detection. First, we used a fast residual attention GAN-based model. Then, we employed the Mixup consistency regularization technique to encourage the discriminator to attend the pixel-level details of the input data. Furthermore, a new cost function for the discriminator is defined based on the mixup to enhance the output. This study is evaluated in the context of lung cancer detection. When compared to empirical optimization, both TPE and Bayesian optimization demonstrate significant improvements in the precision of the fast Residual Attention GAN model used in this study. Various metrics, including precision, f1-score, and the area under the curve (AUC) are employed to assess the system's efficiency.

***Index Terms*—Unsupervised anomaly detection, Hyperparameters optimization, GAN, Mixup.**

I. Introduction

Deep anomaly detection (AD) learning approaches have shown outstanding performance in lung cancer detection [1], [2]. Their powerful learning ability and capability to handle complex patterns make them outperform radiologists in the diagnoses process. Supervised learning is commonly used in lung cancer anomaly detection, but it comes with its own set of challenges. While it is effective, there are several obstacles to overcome. For that, significant efforts have been dedicated to the development of unsupervised models for anomaly detection and localization. Among these deep models, the Generative Adversarial Networks (GANs), have gained popularity [3]. The adversarial relationship between the GAN component networks drives their mutual improvement, leading to the generation of progressively more realistic data. Several GAN-based approaches have been developed for unsupervised anomaly detection of medical images. Among these approaches, authors of [4] proposed a GAN-based model for fast unsupervised detection of COVID-19 anomalies while maintaining a stable training process.

The performance of DL models, including GANs, heavily relies on carefully selecting hyperparameters. This involves determining which hyperparameters to tune, defining suitable value ranges, and selecting optimal values through a meticulous design and experimentation process. Involving domain experts is crucial for effective hyperparameter selection. Automatic design of GAN models is essential for complex architectures with a large parameter space, where exhaustively exploring all combinations is computationally infeasible. Hyperparameter Optimization (HPO) has been extensively researched, including grid search, random search, Bayesian optimization, and gradient-based optimization [5]. Grid and manual search are widely used but have drawbacks when dealing with numerous hyperparameters. Grid search requires predefined values, which is time-consuming and computationally expensive, especially with a large search space. Manual search relies on practitioner intuition, introducing subjectivity and bias [6]. To overcome these limitations and improve the efficiency and reproducibility of HPO, researchers have focused on automating the hyperparameter calibration process. Several advanced techniques have been proposed, such as Bayesian Optimization (BO), Tree-structured Parzen Estimator (TPE), and other sequential model-based optimization algorithms. These methods employ probabilistic models or surrogate functions to guide the search process intelligently, dynamically adapting to the evaluated hyperparameter configurations. By doing so, they can effectively explore the hyperparameter space and allocate more computational resources to promising regions, leading to faster convergence and better performance.

Automated hyperparameter optimization techniques not only relieve the manual search burden but also improve reproducibility. They efficiently fine-tune hyperparameters and uncover overlooked combinations. These automated methods are widely adopted in deep learning, including complex tasks like medical image analysis. This paper focuses on utilizing

979-8-3503-1565-3/23/$31.00 ©2023 IEEE

DOI 10.1109/CW58918.2023.00046

TPE and BO methods to automatically design and train DL models for unsupervised lung cancer detection. The goal is to enhance the classification performance of the GAN model proposed in [4].

This paper is structured as follows: Section. II presents the method used in this paper with the proposed modification and presented the used TPE and BO optimization algorithm. Section. III presents the data and the experimentation setup. Moreover, this section describes the results of the optimized suggested model and elaborates the discussion based on the obtained results. The last section. IV concludes the paper and gives some future perspectives.

II. Proposed Method

A. Model Architecture

The GAN architecture used in this paper is based on a framework introduced in [4]. It comprises three sub-models: a generator G, a discriminator D, and an encoder E. We will discuss each of these sub-models in the following subsections. In this section, we provide an illustration of the overall framework of our approach, as shown in Figure 1.

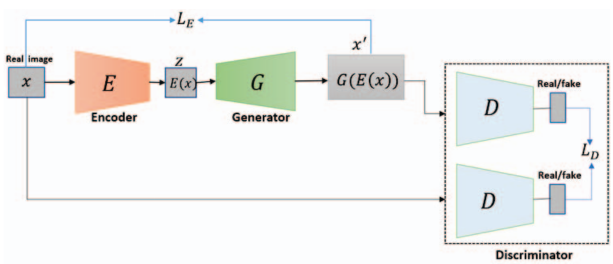

Fig. 1: An overview of the proposed framework. It incorporates an encoder E, a generator G, and a discriminator D.

1) Model Training: We trained our proposed framework only on normal samples for the detection of anomalies. This process involves two learning steps: the GAN training and the encoder training.

Learning step 1: This requires the training of G and D. We adopted the Wasserstein GAN with Gradient Penalty (WGAN-GP) [7] because of its capacity in increasing model stability and performance. The loss function of D L_D is defined as:

$$L_D = \mathbb{E}_{x' \sim \mathbb{P}_g}\left[D\left(x'\right)\right] - \mathbb{E}_{x \sim \mathbb{P}_r}\left[D\left(x\right)\right] + \lambda_g \, \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}}\left[\left(\left\|\nabla_{\hat{x}} D\left(\hat{x}\right)\right\|_2 - 1\right)^2\right] \quad (1)$$

where $D\left(x'\right)$ and $D\left(x\right)$ refer to the discriminator decision at image level for respectively the generated image $x' = G\left(z\right)$ and the real image x. Besides, $\hat{x} = \alpha x + \left(1-\alpha\right) x'$, where $\alpha \in [0,1]$. $\mathbb{P}_r$ and $\mathbb{P}_g$ denote the distributions of respectively real data and generated data. As for the gradient penalty, $\lambda_g = 10$ is the penalty coefficient and $\mathbb{P}_{\hat{x}}$ is the sampling distribution. With regard to the generator, its loss function is defined as follows:

$$L_G = -\mathbb{E}_{x' \sim \mathbb{P}_g}\left[D\left(x'\right)\right]. \quad (2)$$

To train a discriminator effectively, it should focus on semantic changes between real and generated samples, even under class-altering transformations. We utilized the Mixup consistency regularization technique [8] to enhance the discriminator's per-pixel predictions and improve anomaly detection. Following [8], we synthesize a new training sample, x_{mix}, by blending x and $G(z)$ with a random mixing coefficient $\lambda_m \in [0,1]$.

$$x_{mix} = mix\left(x, G\left(z\right), \alpha_m\right) = \alpha_m \, x + \left(1-\alpha_m\right) G\left(z\right). \quad (3)$$

Given the Mixup function in equation. 3, we train the discriminator to provide consistent per-pixel predictions, i.e $[D(mix(x, G(z), \lambda_m))] \approx [mix(D(x), D(G(z)), \lambda_m)]$:

$$L_{D_{mix}} = \|D(mix(x, G(z), \lambda_m)) - mix(D(x), D(G(z)), \lambda_m)\|^2 \quad (4)$$

where $\|.\|$ denotes the $L2$ norm. This consistency loss is then taken between the outputs of D on the Mixup image $[D(mix(x, G(z), \lambda_m))]$ and the Mixup between the outputs of the D on real and fake images $[mix(D(x), D(G(z)), \lambda_m)]$. This loss penalizes the discriminator for producing inconsistent predictions. By doing so, it encourages the discriminator to produce more consistent and reliable predictions, which in turn can improve the overall performance of the GAN model. The new discriminator objective function $L_{D_{new}}$ is as follow:

$$L_{D_{new}} = L_D + \gamma_m . L_{D_{mix}}, \quad (5)$$

where $\gamma_m = 1.0$ controls the strength of the Mixup regularization. The generator objective function L_G remains unchanged, see Eq. 2 Mixup does not change the generator's objective or how it is trained.

Learning step 2: After the GAN training is accomplished, G will be able to map from a latent vector z to a generated image x', $G(z) = x'$. While the encoder E performs the mapping from a real image x to the latent vector z, $E(x) = z$ (see Figure. 1)). In this second learning step, the trained GAN is used with fixed parameters for subsequent encoder training. During the training of E, we minimize the mean squared error (MSE) between the input image x and its associated reconstructed image $G(E(x))$. The encoder's objective function L_E is computed as the sum of the image space loss L_i (Eq. 7) and the discriminator features loss function L_f (see Eq. 8):

$$L_E(x) = L_i(x) + k.L_f(x), \quad (6)$$

where k is a weighting parameter fixed to 1. For L_i, we minimize the mean squared error (MSE) residual loss, of input images x and reconstructed images $G(E(x))$:

$$L_i(x) = \frac{1}{n}\|x - G(E(x))\|^2, \quad (7)$$

where n is the number of pixels in the input image x. The feature matching is carried out thanks to L_f as follows:

$$L_f(x) = \frac{1}{m}\|[D(x)] - D(G(E(x)))\|^2, \quad (8)$$

where m is the dimensionality of the discriminator features representation.

2) Anomaly Scores: Anomaly detection involves identifying patterns in data that deviate from the expected or normal behavior. A higher anomaly score indicates a greater degree of abnormality, while a lower score suggests that the data sample is more typical and adheres to the expected patterns of the dataset. Given a test image x, the image-level anomaly score $A(x)$ is measured as the weighted sum of its reconstruction error and its discrimination error of the discriminator's encoder:

$$A(x) = \frac{1}{n} \|x - x'\|^2 + \frac{1}{m} \|D(x) - D(G(E(x)))\|^2 . \quad (9)$$

B. DL Models Hyperparameters

Among the most powerful unsupervised deep models approaches are the GANs which has been well adopted in the literature for anomaly detection of medical imaging. In this paper, we use the GAN-based model described in section. II-A. **Hyperparameters :** GANs involve a plethora of hyperparameters that impact both the network's architecture and the training process. However, tuning these hyperparameters can be challenging and time-consuming. As a result, there is an increasing demand for automated approaches to hyperparameter calibration in deep learning neural networks. The performance of a GAN is influenced by both training-related hyperparameters (such as learning rate, loss function, mini-batch size, and training iterations) and architectural hyperparameters (including layer sizes, convolution quantities, activation functions, etc.). Table. I provides an overview of the hyperparameters responsible for defining the network's structure and those relevant to optimization and training.

TABLE I: Hyperparameters defining architectures and training process of the GAN and the encoder network.

Hyperparameters	Types	Scope
Optimizer	Categorical/Integer	Adam, RMSProp, Adadelta, SGD
Filter Size	Integer	32, 64, 128,..,1024
Kernel Size	Integer	1, 3, 5, 7
Batch Size	Integer	8, 16, 24, 32, 64
Learning Rate	Float	1e-3, 1e-4, 1e-5, 2e-4, 3e-4, 4e-4
Dropout Rate	Float	0;1
Activation Function	Categorical	ReLU, LeakyReLU, Swish

C. HPO Algorithms

Deep model design requires strong knowledge of algorithms and appropriate hyperparameter optimization techniques. Several methods have been proposed for HPO such as grid search [5], random search, BO [9] and TPE [10]. The TPE and BO success in expensive optimization problems indicates that they may outperform existing methods.

The TPE algorithm is a Sequential Model-Based Optimization (SMBO) approach. In SMBO methods, models are constructed sequentially to approximate the performance of hyperparameters based on past measurements. These models are then utilized to suggest new hyperparameters for evaluation. Therefore, the TPE algorithm follows an iterative process that leverages the historical evaluations of hyperparameters to create a probabilistic model. This model is then employed to recommend the next set of hyperparameters to be evaluated, enhancing the efficiency of the optimization process. To apply the TPE algorithm, let's assume a set of observations $(x^{(1)}, y^{(1))}, ...(x^{(k)}, y^{(k)})$. The TPE algorithm divides the observation results into two categories: good and poor results, based on a pre-defined percentile y^*. The TPE defines the conditional probability density function $p(x|y)$ using the following two equations:

$$p(x|y) = l(x) if y < y^* \quad (10)$$
$$= g(x) if y >= y^* \quad (11)$$

where $l(x)$ is the probability density function formed using the observed variables $x(i)$ such that $y^* > y(i) (= f(x^{(i)})$ and $g(x)$ is the probability density function using the remaining observations. Value y^* is selected to be a quantile γ of the observed y values satisfying $p(y^* > y) = \gamma$. After that, the expected improvement in the acquisition function is reflected by the ratio between the two density functions, which is used to determine the new configurations for evaluation.

BO algorithm aims to minimize a scalar objective function $f(x)$ for a given input x. The output of the algorithm varies depending on whether the function is deterministic or stochastic, even for the same input x. The minimization process involves three main components: a Gaussian process model that represents the objective function $f(x)$, a Bayesian update process that updates the Gaussian model based on new evaluations of the objective function, and an acquisition function $a(x)$. The acquisition function is maximized to determine the next evaluation point. Its purpose is to quantify the expected improvement in the objective function while disregarding values that would increase it. Therefore, the expected improvement (EI) can be calculated as follows:

$$EI(x, Q) = \mathbb{E}_Q \left[\max\left(0, \mu_Q(x_{\text{best}}) - f(x)\right)\right] \quad (12)$$

where Q is the posterior distribution function , x_{best} is the location of the lowest posterior mean and $\mu_Q(x_{best})$ is the lowest value of the posterior mean. Compared to a grid search or manual tuning, BO allows us to jointly tune more parameters with fewer experiments and find better values [11].

III. Experiments and Results

The implementation has been carried out to demonstrate the efficacy of the Hyperparameter Optimization (HPO) algorithm in enhancing the performance of vigilance state classification.

A. Experiment Setting and Data

We performed extensive numerical experiments to evaluate the performance of our proposed model using the ILD datasets [12]. The ILD database is a widely known and publicly accessible resource provided by the University Hospital of Geneva. It consists of 109 High Resolution CT scans of various Interstitial Lung Diseases (ILDs). The CT scans were evaluated by a minimum of three experienced radiologists. The ILD database contains five common research patterns, namely healthy tissue, ground glass, emphysema, fibrosis, and micronodules. In our experiments, we designated healthy tissue as the normal class, while the remaining classes were chosen as abnormal for each individual experiment. It is crucial to emphasize that our training process solely relied on the normal scans for training the model. This architecture is developed using Pytorch whose libraries are written in Python. All experiments were conducted on a Ubuntu 16.04 server with 24Gb memory and a single NVIDIA GeForce RTX 3090Ti GPU. To tackle HPO problems, we use Optuna framework [13], which provides many HPO algorithms including the TPE, and Sherpa framework, which provides BO algorithm.

B. Results and Discussion

We evaluate our model performance with the F1-score, and the area under the AUC. The F1-score is the harmonic mean of Precision and Recall and measures how much the model correctly detect all anomalies while avoiding false ones. The AUC indicates how much the model can distinguish between classes, and a high AUC implies a good result. Table. II presents the hyper-parameter values obtained by the implemented architecture for our model.

TABLE II: Best hyperparameters configurations using TPE and BO algorithms of our model.

Hyperparameters	GAN Values	Encoder Values
Optimizer	Adam	RMSprop
Filter Size	64	64
Kernel Size	3	3
Batch Size	32	32
Learning Rate	2e-4	4e-4
Dropout Rate	0.0	0.5
Activation Function	LeakyReLU	LeakyReLU

Table III presents the precision results of our model, achieved through the utilization of TPE and BO algorithms, along with a comparison to the pre-optimization results. The accuracy of the detection performance is notably high when employing TPE and BO algorithms. Specifically, when using the fast residual attention GAN model, TPE achieves a precision of up to 0.9678, whereas BO achieves a precision of up to 0.9862.

TABLE III: HPO Anomaly Detection Precision of our model.

Models	Without HPO	With TPE	With BO
Our Model	0.9117	0.9678	0.9862

Table IV presents the detection performance of the fast residual attention GAN architecture, which exhibits the best metrics in terms of F1-score and AUC, as shown in Table II. The performance is evaluated for three scenarios: without an optimization process, with the TPE algorithm, and with the BO algorithm.

Given Table III and Table IV, it is evident that incorporating an optimization phase for hyperparameters leads to a significant improvement in the detection performance and implemented deep learning (DL) models. These results highlight the effectiveness of the iterative process involving BO and TPE for our specific application.

TABLE IV: Performance measures of proposed model.

	F1-score			AUC		
	No HPO	TPE	BO	No HPO	TPE	BO
Our model	0.898	0.899	0.904	0.972	0.979	0.987

IV. CONCLUSION

In this paper, we have introduced and investigated the potential of Hyperparameter Optimization (HPO) algorithms to determine the optimal configurations of hyperparameters and enhance the performance of lung cancer detection. The HPO algorithms, namely TPE and Bayesian BO, have been applied to the model to generate the optimal hyperparameter configuration. Our experimental results have demonstrated that the HPO BO method has significantly improved the performance of lung cancer detection compared to an implementation without an optimization process. Additionally, the incorporation of the mixup regularization technique has been found to enhance the obtained results further when compared to a normal model without Mixup.

DECLARATION OF COMPETING INTEREST

The authors state that they have no competing financial interests that could have influenced this paper.

REFERENCES

[1] I. M. Nasser and S. S. Abu-Naser, "Lung cancer detection using artificial neural network," *International Journal of Engineering and Information Systems (IJEAIS)*, vol. 3, no. 3, pp. 17–23, 2019.

[2] A. Shimazaki, D. Ueda, A. Choppin, A. Yamamoto, T. Honjo, Y. Shimahara, and Y. Miki, "Deep learning-based algorithm for lung cancer detection on chest radiographs using the segmentation method," *Scientific Reports*, vol. 12, no. 1, p. 727, 2022.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets, in 'advances in neural information processing systems 27', curran associates," 2014.

[4] N. Nafti, O. Besbes, A. Ben Abdallah, and M. H. Bedoui, "Fast unsupervised residual attention gan for covid-19 detection," pp. 360–372, 2022.

[5] M. Claesen and B. De Moor, "Hyperparameter search in machine learning," *arXiv preprint arXiv:1502.02127*, 2015.

[6] F. F. Firdaus, H. A. Nugroho, and I. Soesanti, "Deep neural network with hyperparameter tuning for detection of heart disease," pp. 59–65, 2021.

[7] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[8] L. H. Nguyen and J.-A. Goulet, "Anomaly detection with the switching kalman filter for structural health monitoring," *Structural Control and Health Monitoring*, vol. 25, no. 4, p. e2136, 2018.

[9] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, 2012.

[10] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in neural information processing systems*, vol. 24, 2011.

[11] M. Injadat, F. Salo, A. B. Nassif, A. Essex, and A. Shami, "Bayesian optimization with machine learning algorithms towards anomaly detection," pp. 1–6, 2018.

[12] A. Depeursinge, A. Vargas, A. Platon, A. Geissbuhler, P.-A. Poletti, and H. Müller, "Building a reference multimedia database for interstitial lung diseases," *Computerized medical imaging and graphics*, vol. 36, no. 3, pp. 227–238, 2012.

[13] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," pp. 2623–2631, 2019.