Journal of the Brazilian Computer Society, 2025, 31:1, doi: 10.5753/jbcs.2025.5177

A Comparative Study of Artificial Intelligence Classification Models for Analyzing Vitiligo Effects in Zebrafish (*Danio rerio*) Images

Pedro Noronha Fagundes [**Research and Development, Unnawave Imaging Diagnostics, Sao Jose dos Campos, Brazil** | ***pedronfagundes@gmail.com***]
Paulo Vitor Costa Silva [**Faculty of Computing, Federal University of Uberlandia. Uberlandia, Brazil** | ***paulo.vitorcs@ufu.br***]
Janina de Sales Guilarducci [**Department of Agronomy/ESAL, Federal University of Lavras, Lavras, Brazil** | ***janina.guilarducci2@estudante.ufla.br***]
Laura Cristina Jardim Porto Pimenta [**Department of Nutrition/FCS, Federal University of Lavras, Lavras, Brazil** | ***laurap@ufla.br***]
Luis David Solis Murgas [**Department of Veterinary Medicine/FZMV, Federal University of Lavras, Lavras, Brazil** | ***lsmurgas@dmv.ufla.br***]
Luciane Vilela Resende [**Department of Agronomy/ESAL, Federal University of Lavras, Lavras, Brazil** | ***luciane.vilela@ufla.br***]
Henrique Fernandes ✉ [**Faculty of Computing, Federal University of Uberlandia. Uberlandia, Brazil** | ***henrique.fernandes@ufu.br***]
Fernando Costa Malheiros [**Research and Development, Unnawave Imaging Diagnostics, Sao Jose dos Campos, Brazil** | ***fernandomalheiros@gmail.com***]

✉ *Faculty of Computing, Federal University of Uberlandia. Uberlandia, 38.408-100, MG, Brazil.*

Received: 08 November 2024 • **Accepted:** 28 June 2025 • **Published:** 11 August 2025

Abstract In recent years, the term Artificial Intelligence (AI) has gained recognition across various fields, such as healthcare, finance, agriculture, and many other sectors due to its versatility and capacity to improve outcome. Within the applications of AI, classification stands out as one of the most prevalent, aiding in optimizing decision-making and efficiently organizing data. In recent years, there has been an increase in the use of zebrafish (*Danio rerio*) in studies related to human dermatological diseases, such as vitiligo, which involves the autoimmune-mediated destruction of the melanocytes in the epidermal layer. Despite the current interest in studies related to this disease, no papers were found applying Machine Learning (ML) or Deep Learning (DL) models to classify the effects of the disease and its treatments. In this context, this paper uses the challenge of evaluating the effects of vitiligo in zebrafish to compare the performance of different AI approaches. The methodology employed in this paper includes image acquisition, dataset creation, preprocessing, model testing, and evaluation. The ML models applied in this study were Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), and Extreme Gradient Boosting (XGBoost), while the DL models included Visual Geometry Group 16 (VGG16) and GoogLeNet. Following the evaluation, SVM and GoogLeNet achieved the best results, correctly classifying 80% and 71% of the data, respectively. Moreover, the former accurately identified all samples in the healthy and treated classes, with misclassifications occurring only within the sick class. The models performed satisfactorily in relation to the objectives of this study and the results exhibited potential for future applications in treating vitiligo in humans.

Keywords: Artificial Intelligence, Classification, Zebrafish, Vitiligo

1 Introduction

In recent years, the term Artificial Intelligence (AI) has gained recognition across various fields, such as healthcare, finance, agriculture, and many other sectors, due to its versatility and capacity to improve outcomes [Zhao *et al.*, 2020]. Due to the rapid technological advances in recent years, AI has become a familiar concept. However, the study and pursuit of algorithms capable of simulating human intelligence began several decades ago. The earliest research efforts aimed at enabling a computer program to learn like the human brain were published in the 1940s [Zhao *et al.*, 2020; Muthukrishnan *et al.*, 2020].

Over the years, the rapid advancement of computational power and the increased availability of data, particularly from the 2000s onward, have enabled AI models to become more complex and capable of being applied to challenging tasks. Thus, new models were developed, and the scope of AI began to expand, enabling a broader range of applications. The AI field is divided into two branches: Machine Learning (ML) and Deep Learning (DL). Both are applied with the aim of performing with better efficiency and accuracy [Toosi *et al.*, 2021].

ML involves algorithms that use statistical methods to learn from data and make predictions based on it. On the other hand, DL is a subset of ML that utilizes Artificial Neural Networks (ANN) to learn from data. The ML has four methods that vary according to the input data: supervised, unsupervised, semi-supervised, and reinforcement learning [Procopio *et al*., 2023; Ongsulee, 2017].

In supervised learning, models are trained using patterns with known classes, and then the predicted value by the model is compared to the correct label, in the learning process. In unsupervised learning, the models have no information about the labels, and thus should be able to classify patterns into classes based on their similarities. Semi-supervised learning involves both labeled and unlabeled patterns, and the models utilize both to learn and generate the outputs. Finally, in reinforcement learning, the models determine the optimal solution through a process of trial and error, where punishments and rewards are used [Ongsulee, 2017].

As mentioned earlier, ML and DL can be successfully applied to many fields, providing satisfactory outputs in various tasks such as classification, speech recognition, regression, and others. Within these applications, classification stands out as one of the most prevalent, aiding in optimizing decision-making and efficiently organizing data [Sharifani and Amini, 2023].

There are several studies focusing on classification problems, explaining new applications, describing new methods, and comparing the performance of different models, among others. Many of these articles address a specific challenge involving the zebrafish (*Danio rerio*) [Sun *et al*., 2020].

This vertebrate organism is commonly employed in studies pertaining to biology, genomics, biomedicine, medicine, and various other fields due to its significant characteristics, which include rapid development and transparency during the early stages [Sun *et al*., 2020]. Another crucial advantage of the zebrafish lies in its genetic similarity, with approximately 70% of its genome overlapping with that of humans [Santoro, 2014; Cestariolo *et al*., 2023].

Thus, in recent years, numerous studies have been published utilizing the zebrafish, including one that trained deep learning networks to classify morphological changes in the zebrafish, generated in response to drug-induced neuronal damage [Ishaq *et al*., 2017]. Another example is a study that used five different ML models and eight types of molecular fingerprints to construct 40 base classifiers, and then constructed an ensemble model with the goal of predicting zebrafish developmental toxicity [Liu *et al*., 2024].

These examples illustrate the versatility of using the zebrafish to study diseases that affect humans and the effects of tested treatments. In recent years, there has been an increase in the use of zebrafish in studies related to human dermatological diseases, such as vitiligo, which involves the autoimmune-mediated destruction of the melanocytes in the epidermal layer [Frantz and Ceol, 2022].

Despite the current interest in studies related to this disease, no papers were found applying ML models to classify the effects of the disease and their treatments. This highlights the potential for such research, which could serve as a basis for future applications of treatments in humans [Frantz and Ceol, 2022].

In this context, this paper uses the challenge of evaluating the effects of vitiligo in zebrafish to compare the performance of different AI models. The images collected were classified into three different categories using six different models [Wang *et al*., 2017].

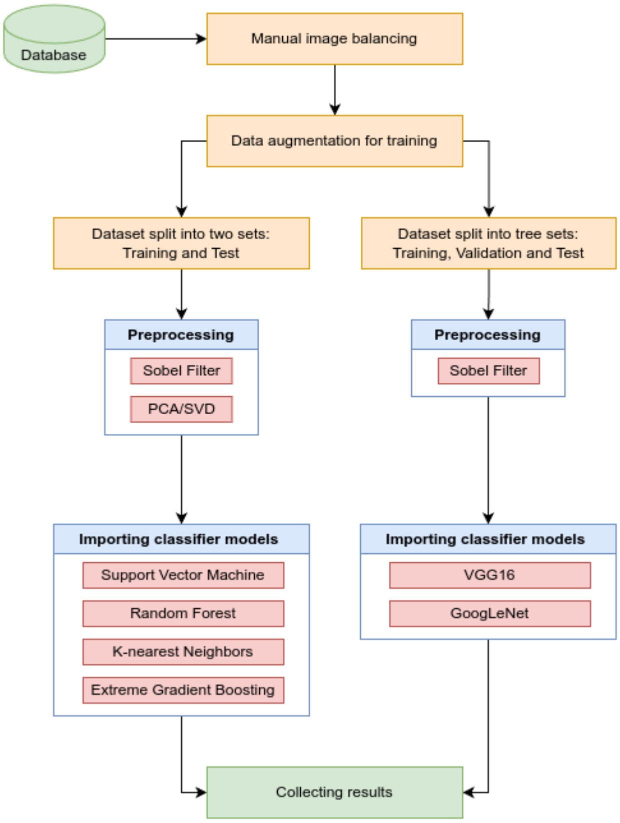
Database
Manual image balancing
Data augmentation for training
Dataset split into two sets: Training and Test
Dataset split into tree sets: Training, Validation and Test
Preprocessing
Sobel Filter
PCA/SVD
Preprocessing
Sobel Filter
Importing classifier models
Support Vector Machine
Random Forest
K-nearest Neighbors
Extreme Gradient Boosting
Importing classifier models
VGG16
GoogLeNet
Collecting results

Figure 1. Flowchart of the research methodology, encompassing image balancing, data augmentation, preprocessing, and results collection.

This study aims to provide insights into both the strengths and limitations of the approaches employed for the classification task. The findings derived from this research could serve as a foundational framework for their application in subsequent studies.

2 Methods

The methodology employed in this study encompasses image acquisition, dataset creation, preprocessing, model testing, and evaluation. The algorithms were implemented using the Python programming language and the Google Colab environment. Figure 1 illustrates the framework developed to compare the strengths and weaknesses of each model in classifying zebrafish.

2.1 Ethical appreciation

The experimental procedures were approved under protocol number 014/2019 by the Animal Experimentation Ethics Committee of the Federal University of Lavras, CEUA-UFLA, Minas Gerais, Brazil. The procedures were carried out in the central vivarium of the Federal University of Lavras, which is located in Lavras, Minas Gerais, Brazil.

2.2 Melanogenesis test in zebrafish larvae

The embryos were obtained from adult zebrafish spawning, from the wild type (WT) strain. The samples were collected in a solution containing 1-phenyl-2-thiourea (PTU, Sigma - Aldrich, St Louis, Missouri, USA) in embryo medium, to inhibit the pigmentation process. After 48 hours of exposure to PTU, the larvae were placed in embryo medium and subsequently exposed to aqueous and ethanolic extracts, obtained through the simple reflux extraction process of the dry matter from *Sonchus oleraceus* leaves, at different harvest times. A total of 5 treatment doses (0.312, 0.156, 0.078, 0.039, 0.020 mg/mL), were defined in triplicate. The embryo medium was renewed daily until 168 hours post-fertilization, totaling 7 days of testing, including 5 days of treatment with the extracts obtained.

2.3 Images acquisition

The larvae were euthanized in tricaine at a concentration of 0.25 g/L [Duarte da Silva *et al.*, 2023], and subsequently maintained in 4% formaldehyde. The images were acquired using an optical microscope (Olen Kasvi®) with an attached video camera, Moticam X5 Plus®. The equipment was calibrated, and the samples were positioned on microscope slides, focusing on the dorsal area, as proposed by [Grisola and Fuentes, 2017]. Additionally, a 4x objective lens was used with high illumination.

A total of 1,320 images were collected, with each of the 22 distinct groups consisting of 60 images, as the experiments were conducted in triplicate. The samples were categorized into three distinct groups: healthy, treated, and sick. The sick group comprised samples with vitiligo, which were induced in the zebrafish. The treated class were categorized with variations based on the age of the *Sonchus oleraceus* plant, the substrate in which the leaf was immersed, and the dosage of the applied medicine.

2.4 Dataset creation

The first step in dataset creation involved selecting images for use in both the training and testing phases. The chosen approach for this study was to classify the zebrafish into three different groups, based on the nature of the available data. Figure 2 illustrates one example from each of the three classes.

These images were randomly selected from the original dataset for the purpose of demonstrating the data utilized in the present study. Figure 2 shows that the groups display certain evident distinctions, such as the number and size of pigmented regions. Nonetheless, few additional visual characteristics are readily perceptible to the human eye, which justifies the use of AI models to accurately classify the groups.

The healthy and sick categories had 60 images each, while the treated had 1,200 available. This unbalanced number could not be used to train the model effectively, as it could negatively impact the learning capacity of the model [Wang *et al.*, 2021].

To address this issue, it was necessary to limit the number of images to 60 by randomly selecting those belonging to the treated class. During this random selection process, it was important to ensure representation from all groups.

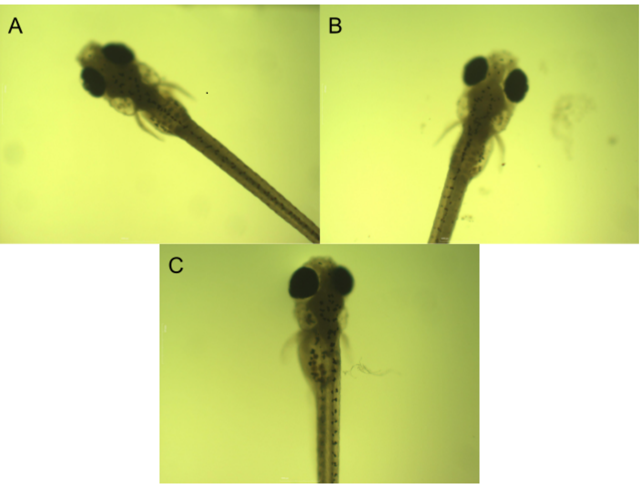

Figure 2. Images exemplifying samples from each of the three classes. A: Sick class. B: Treated class. C: Healthy class.

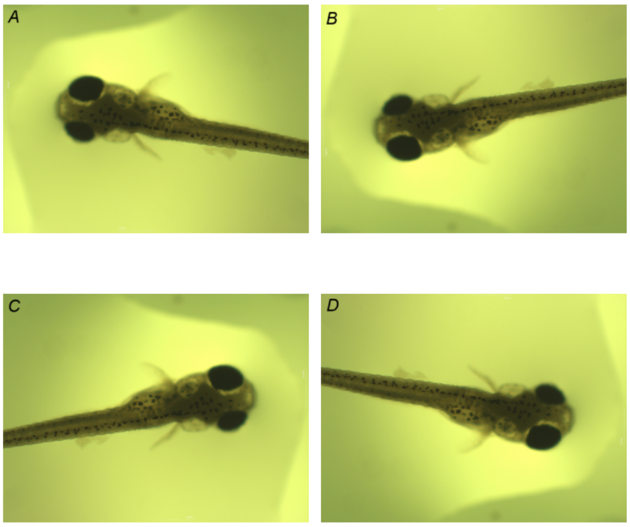

Figure 3. Examples of images obtained after the data augmentation process, which involved flipping the images. A: Original image. B: Image obtained after a vertical flip. C: Image obtained after a horizontal flip. D: Image obtained after both horizontal and vertical flips.

After this step, a total of 180 images were selected, forming a balanced dataset for training and testing the models. The quantity of images can affect the performance of the models, making it impossible for them to achieve good classification accuracy. Therefore, a solution was proposed to increase the sample size.

The technique involves flipping each image around the x-axis, y-axis, and both axes simultaneously. This method results in each image generating three new versions, thus transforming a dataset of 180 images into one containing 720 images. Figure 3 illustrates the results of this technique of data augmentation.

Subsequently, the samples were divided into training and testing sets. It is common to find in the literature a division consisting of 80% for training and 20% for testing, nonetheless, in this research, it was preferable to use the maximum number of images possible for training, given the similarity between the different groups.

Therefore, for the ML approach, the division used consisted of 660 samples for training and 60 for testing. It is important to mention that the division was made to ensure that there were original images in both sets, specifically 165 in the first and 15 in the second.

For the DL approach, the dataset was divided into 60% training, 20% validation, and 20% testing, since this approach required validation samples, resulting in 432, 144, and 144 images, respectively. Although the sets generated for the DL approach were not identical to those created for the ML approach, efforts were made to ensure the presence of original images within each subset, corresponding to 108 original images in the training set, 36 in the validation set, and 36 in the test set. Furthermore, the distribution of original images followed the same data splitting proportions previously mentioned.

This collection of samples was used as input for all algorithms, each requiring a specific format tailored to its own needs. These types and requirements will be further explained in the next section.

2.5 Preprocessing

The classification models have different types of inputs, such as matrices, vectors, or image features, therefore, a preprocessing step was necessary. In this phase, various operations found in the literature were tested with the objective of finding the most appropriate ones for each case.

In this context, literature emphasizes the importance of transforming images into vectors for machine learning-based image classification, where each pixel value represents a feature of the pattern. However, the size of the samples had to be adjusted due to the original image dimensions of 1,944 pixels in height, 2,592 pixels in width, and 3 color channels, resulting in input data with a high number of features [Elkholy and Marzouk, 2024].

This large feature dimensionality made the learning process slow and computationally expensive. Therefore, the images were resized to a height of 400 pixels and a width of 300 pixels, and converted to grayscale.

Some ML algorithms may be sensitive to the scale of input features, such as grayscale images with pixel values ranging from 0 to 255. To address these challenges, all vectors were normalized by dividing the values by 255, resulting in inputs ranging from 0 to 1. This normalization potentially facilitated the learning process.

After these steps, the dataset contained 720 patterns and 120,000 features, which was not sufficient to achieve good results in classification. To improve performance, it was necessary to search for a strategy to supply the models with features that have greater discriminative capacity.

An interesting method involves applying filters capable of reducing noise, enhancing edges, among other functions. After testing several possibilities, it was determined that applying the Sobel filter highlighted certain key points in the images, such as the spots on the zebrafish.

This technique consists of convolving two 3x3 kernels with the original image, one for vertical filtering and the other for horizontal filtering . They can be convolved individually to compute the gradient along each axis, as demonstrated in Equation (1) and Equation (2), or collectively, as demonstrated in this paper, to calculate the gradient magnitude (G), as described in Equation (3) [Anderson *et al.*, 2021; van der Walt *et al.*, 2014].

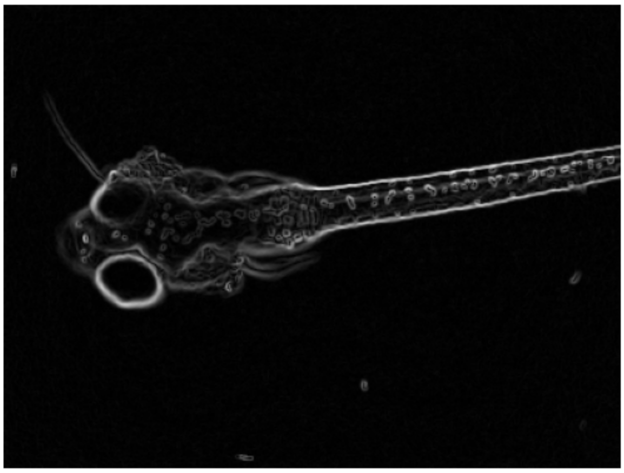

Figure 4. An example of a grayscale image obtained after applying the Sobel filter, highlighting specific key points to enhance the dataset.

$$VerticalEdges = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * OriginalImage \quad (1)$$

$$HorizontalEdges = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * OriginalImage \quad (2)$$

$$G = \sqrt{\frac{VerticalEdges^2 + HorizontalEdges^2}{2}} \quad (3)$$

The new dataset was composed of 720 patterns and 240,000 features, including the original 120,000 features from the grayscale image and an additional 120,000 features resulting from the Sobel filter. Figure 4 illustrates the result of applying the filter in a grayscale image.

Principal component analysis (PCA) is a statistical method used to reduce the dimensions of the input, as not all features added to the model are necessarily important for classification. The method involves employing orthogonal transformation to identify the principal components of the data, which are the components with the highest variance. These components, or directions, are linearly uncorrelated.

To perform the reduction, singular value decomposition (SVD) factorization is applied. This method involves decomposing the original matrix (O) into three others, U, Σ and V^*, as demonstrated in Equation (4). The matrix U is orthogonal, with its columns representing the eigenvectors of OO^*, while Σ is diagonal and contains the singular values of the original dataset. Additionally, the conjugate transpose matrix V^* is also orthogonal with its columns representing the eigenvectors of O^*O [Gerbrands, 1981].

$$O = U\Sigma V^* \quad (4)$$

The methods described above were used in the ML approach, and several tests were conducted with the objective of determining the new dimensionality of the dataset, through analysis of the explained variance by each component. It was concluded that reducing the dataset to 59 components was sufficient to achieve acceptable results in the testing phase. It is important to note that this number was determined using the training dataset to fit the model.

In conclusion of the preprocessing, from the two initial datasets, two new datasets were generated: one with 660 rows and 59 columns, and the other with 60 rows and 59 columns.

In the DL approach, pre-trained Convolutional Neural Networks (CNNs) were chosen, as they have already undergone extensive training on large datasets like ImageNet, resulting in reduced training time and improved performance. The next subsection will describe the models used in both approaches.

2.6 Models

Different algorithms were tested with the objective of comparing their performance in the classification of zebrafish. The models used in the ML approach were: Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN) and Extreme Gradient Boosting (XGBoost). On the other hand, the chosen CNN models were Visual Geometry Group 16 (VGG16) and GoogLeNet.

The selection of the GoogLeNet and VGG16 models was based on their broad relevance in image classification tasks, their well-established performance in the literature, and their computational feasibility for the proposed experiments. Both models are among the most influential in the evolution of convolutional neural networks and are frequently used as references in comparative studies. VGG16 stands out for its simple and standardized architecture, which facilitates the analysis of information propagation through the convolutional layers. In contrast, GoogLeNet incorporates the concept of Inception modules, which optimize the use of computational resources by reducing network complexity without compromising accuracy.

Other convolutional neural network models, such as ResNet50, SqueezeNet, ResNeXt-50, and MNASNet, were also preliminarily evaluated. However, for the continuation of the study, a minimum performance threshold of 60% was adopted. As these models did not reach the required threshold in the tests performed, they were discarded, and the analysis was focused on the VGG16 and GoogLeNet models.

The following subsections provide a description of each of these models.

2.6.1 Support vector machine

SVM is an interesting model for the classification of zebrafish, as it can handle nonlinear relationships, which is the case with the data after applying PCA. Additionally, SVM excels in managing high-dimensional data, which can be a challenging problem. Another significant advantage is its robustness and strong capacity for generalization.

The algorithm, which can be applied to both classification and regression problems, was proposed by Vladimir Vapnik and his coworkers, building upon earlier studies conducted by Vapnik and Lerner, as well as Vapnik and Chervonenkis [Bhavsar and Panchal, 2012; Keleş *et al.*, 2021]. The model identifies the optimal separation between classes by defining a hyperplane that maximizes the margin to the closest data points of each class, known as support vectors. An interesting aspect of SVMs is that they maximize the margin between classes while simultaneously minimizing classification errors.

The SVM can be either linear or nonlinear, depending on the nature of the problem, as some datasets may be separable by a linear hyperplane, while others necessitate more complex solutions. The application of kernel functions enables the resolution of more complex problems by transforming data from a linear space to a high-dimensional feature space, where a hyperplane can effectively separate the samples.

One of the most famous kernel functions is the radial basis function (RBF), which involves mapping the data to a high-dimensional space. The RBF kernel offers advantages over other nonlinear kernels, such as polynomial kernels, due to its requirement of fewer hyperparameters and its computational simplicity.

In the kernel, the *gamma* parameter is fundamental as it determines the influence of support vectors. Higher values of *gamma* result in a shorter radius of influence, while smaller values lead to a wider region of influence by the support vectors.

Some tasks involve multiclass classification, which means that it is not possible to classify the data using only one hyperplane. In such cases, either the one-vs-one or one-vs-rest approaches can be utilized. The key difference is that the one-vs-one approach compares one group against another one at a time, whereas the one-vs-rest approach compares one class against all others at a time.

Another crucial parameter is C because it represents the penalty for misclassified vectors. In this context, if the value of C is higher, the margin will be lower, thereby reducing the classification error during training. Conversely, with a lower value of C, the margin will be larger, potentially resulting in more training errors, however, this can lead to a more generalized model during the testing phase.

2.6.2 Random forest

The random forest algorithm, proposed by Leo Breiman in 2001, is a versatile model capable of handling both small and large samples, as well as high-dimensional spaces [Breiman, 2001]. The model is also capable of minimizing overfitting and achieving high accuracy.

The random forest consists of an ensemble of decision trees that use random subsets of the input data for classification. Decision trees are flowchart-based models that use rules inferred from features to construct a tree, where each tree subsequently contributes a vote for pattern classification.

While decision trees are efficient for certain types of classification tasks, they may not perform well on more complex tasks. This can result in overfitting issues, where the model learns how to classify the training data well but struggles to generalize and correctly classify new or test data.

The randomness inherent in random forests can help mitigate this issue, making the model more suitable for distinguishing between different classes. Weak decision trees are trained with different subsets of the data, and then a majority vote is used to classify the patterns. The different subsets varied in both patterns and features to prevent the trees from making classifications based on the same information.

The use of different trees reduces overfitting and bias, and can also enhance the accuracy of the model. However, a higher number of trees also increases the training time and the required memory.

Therefore, it is important to define key hyperparameters, such as the number of trees, in the random forest, to guarantee optimal performance. Some other hyperparameters are related to the depth of each tree, which is associated with the complexity of the model, while others are linked to the number of samples in the leaf node, as it is possible to consider only the decision nodes with the minimum number of samples.

2.6.3 K-nearest neighbors

A nonparametric algorithm was developed by Evelyn Fix and Joseph Hodges in 1951, subsequently expanded upon by Thomas Cover and Peter Hart in 1967 [Garrido *et al*., 2023; Cover and Hart, 1967]. This algorithm is now recognized as k-nearest neighbors.

The positive aspects of the model include its simplicity and ease of implementation, as well as its versatility for both classification and regression problems. As a consequence, despite being an algorithm developed many years ago, it is still widely utilized in literature and as a solution for various companies

KNN is considered a lazy learner since it does not perform any generalization on the training data, but rather learns the entire dataset. Then, to classify new patterns, the model measures the distances between them and their nearest neighbors.

The number of neighbors considered is represented by “k”, which is a fundamental hyperparameter for classification, as the model’s performance is sensitive to this value. A low value of neighbors may result in poor performance when classifying new values, while larger choices for “k” may cause the classifier to overlook smaller details.

Another important hyperparameter is the metric used to compute the distance between the pattern and its neighbors. This parameter is important because it will define the nearest neighbors used in the classification. Some of the more utilized distances are Euclidean, which is the square root of the sum of the squares of the differences between the coordinates, and Manhattan, defined as the sum of the absolute differences between the points in all dimensions. It is important to note that, as KNN measures distances to neighbors, larger datasets may lead to expensive calculations, which represents a limitation of the algorithm.

The decision-making in KNN occurs through voting, where the pattern is assigned to the class that has the most neighbors in its proximity. In this process, closer neighbors have a higher weight in the decisions compared to distant ones.

2.6.4 Extreme gradient boosting

XGBoost, proposed by Tianqi Chen, has become a widely recognized model in the literature, known for its ability to achieve state-of-the-art performance in various classification tasks [Chen and Guestrin, 2016]. Due to this capability, it is utilized in many ML competitions as well as in real-life problems for classification and prediction.

The algorithm has a significant advantage over other models, primarily due to its scalability across various scenarios, as it can scale to datasets with billions of values and runs ten times faster than other models. Another important characteristic of XGBoost is its capability to handle datasets with missing values effectively.

The algorithm model resembles a random forest in that both utilize decision trees, however, XGBoost has the advantage of being an implementation of gradient boosted decision trees. This technique corrects errors from previous predictors by training subsequent ones to compensate for those errors.

Since the model uses decision trees, some hyperparameters of XGBoost are related to them, such as the maximum depth and the number of parallel trees, which are constructed during each iteration. These parameters affect the complexity of the model.

Many other hyperparameters affect performance, such as the learning rate, which is related to the step size shrinkage used after the boosting step. Additionally, there are regularization parameters, lambda and alpha, which affect how conservative the model will be. These are just some examples, as XGBoost has many other parameters that can be adjusted to achieve better results.

The voting process to define the class takes into consideration the weights of each predictor. The weights are attributed based on the tree’s ability to correct the errors of the previous ones, ensuring that predictors with better performance have greater influence.

2.6.5 GoogLeNet (Inception)

GoogLeNet, also known as Inception v1, is a CNN introduced in 2014 by the Google Research team in the paper [Szegedy *et al*., 2015]. The authors sought an innovative approach to increase the depth and width of neural networks while maintaining computational efficiency. The emergence of GoogLeNet marked a turning point in the field of computer vision, standing out for its modular architecture and efficient use of computational resources.

The main goal of GoogLeNet was to improve the accuracy of image recognition tasks, such as the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC), while addressing issues of overfitting and high computational demand. The Google team recognized that simply increasing the depth and complexity of networks was not a sustainable solution, as it would lead to exponential growth in resource consumption. Thus, they proposed a new architecture that could explore multiple convolution scales simultaneously without a significant increase in computational complexity.

The architecture of GoogLeNet is composed of a set of modules called “Inception Modules”, which drastically re-

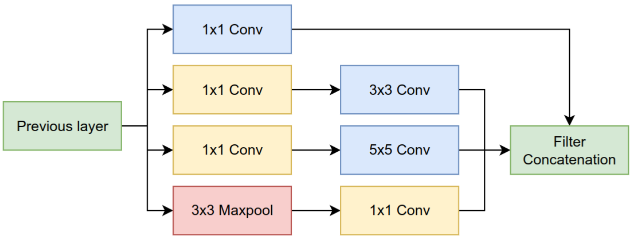

Figure 5. Flowchart related to the Inception-v1 version implemented in the presented work.

duce the number of parameters in the network to 4 million, compared to AlexNet's 60 million. Each Inception Module is designed to perform convolutions of different sizes (1x1, 3x3, 5x5) and pooling operations in parallel, allowing the network to extract a wide range of features from different spatial scales. Additionally, 1x1 convolutions are used to reduce the dimensionality of the data before applying larger convolutions, which helps decrease computational cost. This approach enables the network to be both deep and wide, efficiently capturing complex image features [Szegedy *et al.*, 2015].

Among the various versions of CNNs that incorporate the Inception Module, the Inception-v1 and Inception-v3 versions are models available in the open-source ML library PyTorch. The present work utilizes Inception-v1, as illustrated by Figure 5.

2.6.6 Visual Geometry Group 16

The VGG16, one of the most popular variants of the VGG network series, was introduced in [Simonyan and Zisserman, 2014]. This work aimed to explore the impact of the depth of CNNs on the accuracy of image recognition tasks. At the time, the trend was to increase the complexity of networks to improve performance, and the authors sought a design that could maximize this depth without compromising the simplicity and efficiency of the network.

The main goal of VGG16 was to improve accuracy in image recognition tasks, and to achieve this, the authors designed an architecture that emphasized network depth by using many stacked convolutional layers. VGG16 has 16 weighted layers, including 13 convolutional layers and 3 fully connected layers. The choice of multiple consecutive convolutional layers allowed the network to learn complex and high-level features of images, resulting in superior performance in classification tasks.

The architecture of VGG16 is characterized by the consistent use of small 3x3 convolutions with a stride of 1 and 2x2 pooling operations with a stride of 2. All convolutional layers use 3x3 filters, which are small enough to capture simple patterns but, when stacked, can capture more complex features. The network starts with 64 filters in the initial layers, and after each pooling operation, the number of filters is doubled, reaching 512 in the deeper layers. This modular design facilitates the training and implementation of the network [Simonyan and Zisserman, 2014].

A distinctive feature of VGG16 is its simplicity. Unlike previous architectures that used filters of different sizes and varied configurations, VGG16 adopts a homogeneous ap-

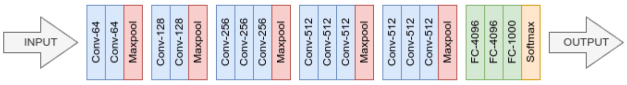

Figure 6. Illustration of the VGG16 architecture applied to the classification task.

proach, using only 3x3 convolutions. This consistency in architecture not only simplifies the network design but also improves computational efficiency, making it more suitable for training and inference on the hardware available at the time. Additionally, the uniformity of the network facilitates knowledge transfer to other tasks and the adaptation of the architecture to different computer vision problems. Figure 6 blueillustrates the architeture of VGG16.

2.7 Hyperparameters optimization

As explained, each of the models presented possesses numerous hyperparameters that can be adjusted to attain optimal results. In this manner, it was necessary to employ a strategy to find the optimal combination of parameters.

There are several strategies to find these combinations, ranging from random search to algorithms specifically tailored for this kind of task. A common method applied is grid search, which conducts a search over all desired parameter values [Liashchynskyi and Liashchynskyi, 2019].

Since the models have various hyperparameters and values that can be selected, grid search can be an exhaustive method. Thus, it was important to define a search space to focus the searches on the previously mentioned hyperparameters, as they could directly affect the results of this study.

To identify the best results and, consequently, determine the optimal values, various metrics can be utilized, individually or in combination, such as accuracy, F1-score, and precision, among others. All of these metrics will be explained in detail in the following subsection, in the context of evaluating the model's performance.

2.8 Metrics

The metrics used to evaluate the models were some of the widely accepted standards, as they are among the most commonly referenced and utilized in literature. Accuracy, precision, recall, and F1-score were the chosen metrics, along with the confusion matrix used to understand the performance of the classifier [Handelman *et al.*, 2019; Mahoro and Akhloufi, 2024].

The confusion matrix consists of three rows, denoting the actual classes, and three columns, denoting the predicted classes. Therefore, the principal diagonal consists of the images correctly classified. These values could be used individually or simultaneously to evaluate the results.

When there are only two classes, the confusion matrix is actually a 2 by 2 matrix composed of the correctly classified values, true positives (TP) and true negatives (TN), along the principal diagonal. The other two positions represent the values of the positive class classified as negatives, false negatives (FN) and the negative values classified as positives, false positives (FP).

Accuracy (acc) can be defined as the ratio of correct predictions made by the model. In other words, it is the division of the number of correct predictions by the total number of predictions, as illustrated in Equation (5).

$$acc = \frac{CorrectPredictions}{Predictions} \tag{5}$$

Precision is the proportion of positive values correctly classified. It is defined as the ratio of true positive values to the sum of true positives and false positives, illustrated in Equation (6). Since this paper addresses multiclass classification, the measured precision is actually the average of the precision of each class, as presented in Equation (7).

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$AvgPrecision = \frac{Precision_0 + Precision_1 + Precision_2}{3} \tag{7}$$

Recall, which is equal to the true positive ratio or sensitivity, measures the positive predictions among all positive values, including those misclassified, as shown in Equation (8). Since the challenge involves multiclass classification, the recall was calculated as the average across all labels, as presented in Equation (9).

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$AvgRecall = \frac{Recall_0 + Recall_1 + Recall_2}{3} \tag{9}$$

The F1-score can be defined as the harmonic mean of precision and recall, as illustrated in Equation (10). As mentioned earlier, the chosen approach to evaluate the models in this work involved calculating the mean, as presented in Equation (11).

$$F1 = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{10}$$

$$AvgF1 = \frac{F1_0 + F1_1 + F1_2}{3} \tag{11}$$

3 Results

As previously mentioned, the objective of this study was to compare the performance of different AI approaches in order to identify the strengths and weaknesses of each in the context of this specific classification task. This challenge involves the available images of zebrafish and the specific dosage used to treat some of the sick animals.

The hyperparameters defined for each of the six models utilized are illustrated in Table 1. These values were defined using the training dataset, with accuracy being the primary focus in determining the optimal values during hyperparameter optimization.

Using these hyperparameters, the models were trained and subsequently tested with different images. During this process, evaluation metrics were computed for both the ML and DL models.

Table 1. Values defined after the hyperparameter optimization phase.

Algorithms	Hyperparameters
SVM	C: 5 kernel: rbf gamma: scale decision function shape: ovo class weight: None random state: 42
RF	n estimators: 500 max depth: None min samples split: 9 min samples leaf: 2 random state: 42
KNN	n neighbors: 30 weights: uniform algorithm: auto leaf size: 30 p: 1
XGBoost	max depth: 15 eta: 0.1 objective: multi:softmax num class: 3 num parallel tree: 3 eval metric: auc
VGG16	lr = 0.000001 weight decay = 0.0001
GoogLeNet	lr = 0.00001 weight decay = 0.0001

It is important to emphasize that the results were analyzed separately for each AI branch, due to differences in the number of training and testing samples across them. Although the results are displayed jointly in the following tables, the analyses were performed separately to better identify the strengths and weaknesses of each model and branch.

The models trained with the hyperparameters were saved to be tested later with the test dataset. This was done to ensure consistent results when the same test data was utilized. This is necessary because some models can generate different results when run at different times due to their inherent randomness.

The results for accuracy, recall, F1-score, and precision obtained during the test phase for all models are summarized in Table 2. These results facilitate the comparison of overall performance and highlight key findings.

4 Discussion

Regarding the machine learning models, RF and KNN demonstrated comparable overall performance, with results variations between 1% and 2%. These two models show the lowest performance, with values ranging from 0.62 to 0.65.

Table 2. Results of all metrics calculated for each of the models tested.

	SVM	RF	KNN	XGBoost	VGG16	GoogleNet
accuracy	0.80	0.63	0.65	0.67	0.60	0.71
recall	0.80	0.63	0.65	0.67	0.60	0.71
F1-score	0.77	0.62	0.63	0.65	0.60	0.71
precision	0.85	0.62	0.64	0.67	0.61	0.74

Among these two, KNN exhibited the best performance, with accuracy and recall reaching 0.65. This suggests that KNN was slightly more effective in correctly identifying the samples and classifying the positive class compared to the other models. This result is further supported by the F1-score, which was also marginally better.

XGBoost achieved slightly better results than the two models mentioned, with an accuracy of 0.67. The equal results of recall and precision indicate that the number of erroneously classified values was equal between false positives and false negatives. The technique of correcting errors from previous predictors by training subsequent ones to compensate for those errors proved to be more suitable for the dataset of the present research, when compared to the RF model, which is also based on decision trees.

Notably, the SVM achieved the best results among both the ML and DL branches, correctly identifying 80% of the data. In terms of precision, the model achieved a score of 0.85, which was the best result among all metrics utilized and models tested. In this way, SVM demonstrated superior performance in classifying the positive class. The ability of SVM to handle nonlinear relationships by mapping data to a high-dimensional space using the RBF kernel may explain the strong performance achieved by the model, which demonstrated better adaptation to the dataset employed.

Among the DL models, it is evident that GoogLeNet demonstrates superior overall performance, achieving results significantly better than those obtained by VGG16. This conclusion can be drawn, as the difference between the values obtained in all four metrics exceeded 10%.

The VGG16 demonstrated a limited capacity for correctly identifying the classes, with an accuracy of 0.60. The highest value for this model was in precision, which was only 61%. In fact, these were the lowest results observed across all the models tested.

On the other hand, GoogLeNet exhibited an acceptable ability to differentiate between the classes, with accuracy, recall, and F1-score all achieving a value of 71%. The model achieved its best value in precision, which was 0.74, highlighting its ability to correctly identify the positive class.

The results presented so far represent the average performance across each class, providing an overall understanding of the models' performance. To analyze the classification of each sample in a specific way, it is necessary to examine the confusion matrix. Table 3 summarizes all the confusion matrices obtained in the test phase. It is important to emphasize that the rows represent the actual class of the pattern and the columns represent the predicted classes.

Table 3. Confusion matrices of all six models obtained in the test phase.

	SVM			RF			KNN		
healthy	20	0	0	10	2	8	12	0	8
treated	0	20	0	0	20	0	1	19	0
sick	4	8	8	12	0	8	4	8	8
	XGBoost			VGG16			GoogleNet		
healthy	16	4	0	10	9	29	7	9	32
treated	0	16	4	8	32	8	8	40	0
sick	8	4	8	27	12	9	31	14	3

An important observation derived from Table 3 is that all models showed good performance in identifying the treated class, with some of them correctly classifying all samples of this class. In fact, despite the RF not presenting good accuracy, it managed to correctly classify all samples of this class. Both VGG16 and GoogLeNet demonstrated better performance in identifying this class, successfully classifying the majority of the samples.

The healthy class patterns were also identified with good performance in the overall assessment. However, the ML models were more capable of correctly classifying this class, while the DL models erroneously classified most of the samples as belonging to the sick class.

At least, it is noteworthy that all classifiers struggle to identify the sick class. For the ML models, both RF and XGBoost made some erroneous classifications by predicting that samples from this class were, in fact, healthy. On the other hand, SVM and KNN predominantly misclassified sick samples as treated. In fact, SVM only exhibited errors in the classification of sick samples, accurately categorizing all patterns associated with the other classes. Regarding the DL models, both classified the majority of the sick class samples as healthy, correctly identifying only a small proportion of them.

A detailed analysis of Tables 2 and 3 indicates that the ML models exhibited strong performance in distinguishing between treated and healthy classes, particularly in identifying the former. A similar conclusion can be drawn from the analysis of the DL models; however, in this case, the performance was significantly better for the treated samples, while the models struggled to classify the healthy ones.

One possible explanation for the results obtained during the testing phase suggests that the sick samples may exhibit greater unpredictability or less distinguishable features, making them more difficult to classify accurately. In contrast, although some variability exists among the healthy samples, they may possess more consistent and distinguishable features, such as the size and number of pigmented regions, which facilitates their identification by ML models. However, these features were not as clearly recognized by the DL models, as both types of models showed some confusion in classifying the healthy and treated classes.

Regarding the treated class, both the ML and DL models demonstrated satisfactory performance. Despite the administration of different doses, the samples may have exhibited distinguishing features that set them apart from both the diseased and healthy classes.

An important observation regarding this analysis is that the ML models were trained and tested using the principal components selected after the application of PCA, whereas this was not performed for the DL models. Thus, the original features were not directly used by the ML models; instead, the principal components derived from these features were. This may potentially explain the superior performance exhibited by this group of algorithms, as these components could have provided more relevant information to the models.

5 Conclusion

In the context of this study, which aimed to compare the performance of different AI models in classifying healthy, treated, and sick zebrafish samples related to vitiligo, the comparative analysis revealed that GoogLeNet outperformed VGG16, achieving an accuracy of 0.71 and a precision of 0.74. GoogLeNet was particularly effective in classifying the treated class but struggled to correctly identify the healthy and sick classes. Conversely, SVM demonstrated the best overall performance among all ML models tested, with an accuracy of 0.80 and a precision of 0.85, successfully classifying all samples from the healthy and treated classes.

Although some DL models showed promise, SVM remains the most robust choice for this specific classification task. This outcome may be attributed to the use of the RBF kernel function, which may have been more suitable for the characteristics of the dataset.

Considering the difference in the number of samples used for training and testing in the two approaches, the objective of the study was not to directly compare the performance of the respective models. However, it is important to note that when efficient ML techniques are applied, it is possible to achieve results comparable to, and in some cases, superior to those obtained with DL models. This highlights the importance of carefully selecting the model, taking into account factors such as problem complexity, data availability, and computational efficiency, rather than assuming that more sophisticated approaches will always yield better performance.

Overall, the models performed satisfactorily given the objectives of this study, which suggests opportunities for future improvements through new preprocessing techniques and testing with additional models. It is also important to emphasize that the results were obtained using a single dataset split for both approaches. Therefore, more comprehensive studies should be conducted in the future involving different combinations of training, validation, and testing images.

This study proved to be relevant by using images of zebrafish to detect the presence of vitiligo. The choice of zebrafish as a study model is very promising, given the increased use of this organism in research related to the treatment of human dermatological diseases, yielding good results. Consequently, the study not only contributes to advancing the understanding of vitiligo but also demonstrates the potential application of these methods in future human treatments, highlighting the importance of continuing to explore this line of research.

Declarations

Funding

This work was supported by grant number 2022/08103-2, São Paulo Research Foundation (FAPESP), by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 001, and by the National Council for Scientific and Technological Development (CNPq) – Finance Code 312530/2023-4.

Authors' Contributions

PF: Methodology, Software, Formal analysis, Data Curation, Writing - Original Draft & Project administration. **PS:** Methodology, Software, Formal analysis, Data Curation & Writing - Original Draft. **JG:** Validation, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft & Funding acquisition. **LP:** Validation, Investigation, Resources & Writing - Review & Editing. **LM:** Validation, Investigation, Resources & Writing - Review & Editing. **LR:** Validation, Investigation, Resources & Writing - Review & Editing. **HF:** Conceptualization, Methodology, Writing - Review & Editing, Supervision & Funding acquisition. **FM:** Conceptualization, Methodology, Writing - Review & Editing, Supervision & Funding acquisition.

Competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Availability of data and materials

The dataset used in this study is not publicly available. It may be made available by the authors upon request.

References

Anderson, L. S., Armstrong, W. H., Anderson, R. S., and Buri, P. (2021). Debris cover and the thinning of kennicott glacier, alaska: in situ measurements, automated ice cliff delineation and distributed melt estimates. *The Cryosphere*, 15(1):265–282. DOI: 10.5194/tc-15-265-2021.

Bhavsar, H. and Panchal, M. H. (2012). A review on support vector machine for data classification. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, 1(10):185–189. Available at:`https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=d683a971524a0d76382ce335321b4b8189bc8299`.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32. DOI: 10.1023/A:1010933404324.

Cestariolo, L., Luraghi, G., L'Eplattenier, P., and Matas, J. F. R. (2023). A finite element model of the embryonic zebrafish heart electrophysiology. *Computer Methods and Programs in Biomedicine*, 229:107281. DOI: 10.1016/j.cmpb.2022.107281.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. DOI: 10.1145/2939672.2939785.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27. DOI: 10.1109/tit.1967.1053964.

Duarte da Silva, K. C., do Carmo Rodrigues Virote, B., de Fátima Santos, M., Dias Castro, T. F., de Azevedo Martins, M. S., Carneiro, W. F., and Solis Murgas, L. D. (2023). Embriotoxic and antioxidant effects of cymbopogon citratus leaf volatile oil on zebrafish. *Revista Brasileira de Farmacognosia*, 33(4):778–789. DOI: 10.1007/s43450-023-00410-w.

Elkholy, M. and Marzouk, M. A. (2024). Deep learning-based classification of eye diseases using convolutional neural network for oct images. *Frontiers in Computer Science*, 5:1252295. DOI: 10.3389/fcomp.2023.1252295.

Frantz, W. T. and Ceol, C. J. (2022). Research techniques made simple: zebrafish models for human dermatologic disease. *Journal of Investigative Dermatology*, 142(3):499–506. DOI: 10.1016/j.jid.2021.10.016.

Garrido, I., Lecube, J., Mzoughi, F., Aboutalebi, P., Ahmad, I., Cayuela, S., and Garrido, A. (2023). A machine-learning approach for prognosis of oscillating water column wave generators. In *2023 27th International Conference on Circuits, Systems, Communications and Computers (CSCC)*, pages 1–7. IEEE. DOI: 10.1109/CSCC58962.2023.00009.

Gerbrands, J. J. (1981). On the relationships between svd, klt and pca. *Pattern recognition*, 14(1-6):375–381. DOI: 10.1016/0031-3203(81)90082-0.

Grisola, M. A. C. and Fuentes, R. G. (2017). Phenotype-based screening of selected mangrove methanolic crude extracts with anti-melanogenic activity using zebrafish (danio rerio) as a model. *ScienceAsia*, 43(3). DOI: 10.2306/scienceasia1513-1874.2017.43.163.

Handelman, G. S., Kok, H. K., Chandra, R. V., Razavi, A. H., Huang, S., Brooks, M., Lee, M. J., and Asadi, H. (2019). Peering into the black box of artificial intelligence: evaluation metrics of machine learning methods. *American Journal of Roentgenology*, 212(1):38–43. DOI: 10.2214/ajr.18.20224.

Ishaq, O., Sadanandan, S. K., and Wählby, C. (2017). Deep fish: deep learning–based classification of zebrafish deformation for high-throughput screening. *SLAS Discovery: Advancing Life Sciences R&D*, 22(1):102–107. DOI: 10.1177/1087057116667894.

Keleş, S., Günlü, A., and Ercanli, İ. (2021). Estimating aboveground stand carbon by combining sentinel-1 and sentinel-2 satellite data: a case study from turkey. In *Forest Resources Resilience and Conflicts*, pages 117–126. Elsevier. DOI: 10.1016/b978-0-12-822931-6.00008-3.

Liashchynskyi, P. and Liashchynskyi, P. (2019). Grid search, random search, genetic algorithm: a big comparison for nas. *arXiv preprint arXiv:1912.06059*. http://dx.doi.org/10.48550/arXiv.1912.06059. DOI: 10.48550/arXiv.1912.06059.

Liu, G., Li, X., Guo, Y., Zhang, L., Liu, H., and Ai, H. (2024). Ensemble multiclassification model for predicting developmental toxicity in zebrafish. *Aquatic Toxicology*, page 106936. DOI: 10.2139/ssrn.4776693.

Mahoro, E. and Akhloufi, M. A. (2024). Breast cancer classification on thermograms using deep cnn and transformers. *Quantitative InfraRed Thermography Journal*, 21(1):30–49. DOI: 10.1080/17686733.2022.2129135.

Muthukrishnan, N., Maleki, F., Ovens, K., Reinhold, C., Forghani, B., Forghani, R., *et al.* (2020). Brief history of artificial intelligence. *Neuroimaging Clinics of North America*, 30(4):393–399. DOI: 10.1016/j.nic.2020.07.004.

Ongsulee, P. (2017). Artificial intelligence, machine learning and deep learning. In *2017 15th international conference on ICT and knowledge engineering (ICT&KE)*, pages 1–6. IEEE. DOI: 10.1109/ICTKE.2017.8259629.

Procopio, A., Cesarelli, G., Donisi, L., Merola, A., Amato, F., and Cosentino, C. (2023). Combined mechanistic modeling and machine-learning approaches in systems biology–a systematic literature review. *Computer methods and programs in biomedicine*, 240:107681. DOI: 10.1016/j.cmpb.2023.107681.

Santoro, M. M. (2014). Zebrafish as a model to explore cell metabolism. *Trends in Endocrinology & Metabolism*, 25(10):546–554. DOI: 10.1016/j.tem.2014.06.003.

Sharifani, K. and Amini, M. (2023). Machine learning and deep learning: A review of methods and applications. *World Information Technology and Engineering Journal*, 10(07):3897–3904. Available at:`https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4458723`.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. DOI: 10.48550/arXiv.1409.1556.

Sun, M., Yang, X., and Xie, Y. (2020). Deep learning in aquaculture: A review. *J. Comput*, 31(1):294–319. DOI: 10.3966/199115992020023101028.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9. DOI: 10.1109/cvpr.2015.7298594.

Toosi, A., Bottino, A. G., Saboury, B., Siegel, E., and Rahmim, A. (2021). A brief history of ai: how to prevent another winter (a critical review). *PET clinics*, 16(4):449–469. DOI: 10.1016/j.cpet.2021.07.001.

van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., Yu, T., and the scikit-image contributors (2014). scikit-image: image processing in python. *PeerJ*, 2:e453. DOI: 10.7287/peerj.preprints.336v1.

Wang, J., Perez, L., *et al.* (2017). The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Networks Vis. Recognit*, 11(2017):1–8. DOI: 10.48550/arxiv.1712.04621.

Wang, L., Han, M., Li, X., Zhang, N., and Cheng, H. (2021). Review of classification methods on unbalanced data sets. *IEEE Access*, 9:64606–64628. DOI: 10.1109/access.2021.3074243.

Zhao, S., Blaabjerg, F., and Wang, H. (2020). An overview of artificial intelligence applications for power electronics. *IEEE Transactions on Power Electronics*, 36(4):4633–4658. DOI: 10.36227/techrxiv.12431081.