ANERIS

Operational Sensing Life Technologies for Marine Ecosystems

`

Deliverable D4.3 – AMOVALIH code and documentation

Lead Beneficiary: CSIC

Author/s: Andreu Fornos, Ivan Rodero, Xavier Salvador, Berta Companys.

29/12/2023

Funded by
the European Union

Prepared under contract from the European Commission

Grant agreement No. 101094924

EU Horizon Europe Research and Innovation action

Project acronym:	**ANERIS**
Project full title:	**operAtional seNsing lifE technologies for maRIne ecosystemS**
Start of the project:	January 2023
Duration:	48 months
Project coordinator:	Jaume Piera
Deliverable title:	AMOVALIH code and documentation
Deliverable n°:	D4.3.
Nature of the deliverable:	Other
Dissemination level:	Public
WP responsible:	WP4
Lead beneficiary:	CSIC
Citation:	Fornos, A., Rodero, I., Salvador, X., Companys, B. (2023). AMOVALIH code and documentation. Deliverable D4.3 y EU Horizon Europe ANERIS Project, Grant agreement No. 101094924
Due date of deliverable:	Month n°12
Actual submission date:	Month n°12

Deliverable status:

Version	Status	Date	Author(s)
0.1	Initial draft	2/12/2023	QUANTA
0.2	Complete draft	28/12/2023	QUANTA + CSIC
0.3	Internal review	29/12/2023	CSIC
1.0	Final	29/12/2023	QUANTA + CSIC

Table of Contents

Preface

This document is a deliverable for the ANERIS project, funded under the European Union's Horizon Europe Research and Innovation Action under grant agreement No. 101094924.

The Advanced Marine Observations VALidation-Identification system based on Hybrid intelligence (AMOVALIH) will be developed in a similar way to the previous technologies. The code will be integrated into the Citizen Observatory MINKA to facilitate advanced services for identifying and validating reported observations. The code will be shared in open (Git-based) repositories for future improvements and in the EOSC Marketplace as well. The final code version, with its linked documentation will be made public upon its release.

Summary

One of the major bottlenecks in artificial intelligence (AI) is that labeling training data, and validating model outputs still require substantial human supervision and work, which is why many AI applications rely on the same training data sets and do not often closely enough inspect the outcomes. Hybrid Intelligence, which combines human and Artificial Intelligence to achieve superior results collectively, could overcome the challenge of obtaining training sets and learning much more effectively. Contributions to the project: AMOVALIH will be a hybrid intelligent system to classify images of marine life, combining reputation-based classification systems based on the contributions from humans, combined with advanced automatic identification systems that will participate as virtual agents and learn progressively with new validated observations from new species. This will for operational marine life monitoring be the first time that one approach of hybrid intelligence is developed for marine life monitoring. Validation Case Studies: CS3, CS4.

This document describes the initial implementation of AMOVALIH, including a novel hierarchical AI model, a robust API that can integrate multiple AI models and human interaction towards the targeted hybrid intelligence approach.

List of Abbreviations

AI - Artificial intelligence

AI HLEG - High-level expert group on artificial intelligence

ANN - Artificial Neural Network

APF - “Animalia”, “Plantae”, and “Fungi” categories

API	Application Programming Interface
AUC	area under the curve
BoF	Bag of Features
CNN	Convolutional Neural Networks
DL	Deep Learning
EGI	European Grid Initiative
FPR	false positive rate
GBIF	Global Biodiversity Information Facility
GLCM	Gray-Level Co-occurrence Matrix
KNN	K-Nearest Neighbors
ML	Machine learning
MDP	Markov Decision Process
NoSQL	non-relational types of databases
PCA	Principal Component Analysis
PC	“Protozoa” and “Chromista” categories
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RFE	Recursive Feature Elimination
ROC	Receiver Operating Characteristic
SDG	Sustainable Development Goals
SVM	Support Vector Machine
TPR	true positive rate

1. Introduction

The ongoing loss of biodiversity presents a pressing global issue, threatening our planet's ecosystem and its life-sustaining capabilities (Tilman, 2000). This predicament has drawn significant concern from both the scientific community and the broader public, making the analysis of biodiversity and environmental indicators a topic of paramount importance (Sala et al., 2000). Citizen observatories, such as MINKA, play a crucial role in promoting citizen engagement in marine ecosystem sustainability while generating an extensive dataset of flora and fauna images that require expert classification.

Traditionally, the assessment and study of biodiversity and environmental health have been reliant on labor-intensive, costly, and often inefficient manual monitoring methods (Bartkowski, Lienhoop, and Hensjürgens, 2015). AMOVALIH aims to address this by leveraging advances in AI, with a focus on blended AI techniques. We aim to analyze large datasets of bioimages, utilizing expert knowledge in the field and data science techniques.

While numerous approaches exist for generating bioimage classification models, we acknowledge that it is unrealistic to expect that our generated AI models can outperform all others in classifying all species (Nanni et al., 2019). Therefore, our proposed solution is to create an API allowing users to select from a range of AI models to accurately classify their bioimages. This API will serve as an "API of APIs", where developers and service providers with endpoints containing AI models for classifying images can register their endpoints to our API, thereby making their model accessible to our community (López García, 2019), and continuously improving their models by leveraging the proposed hybrid approach.

Our contribution lies in its ability to enhance the precision and scale of monitoring and understanding ecosystem health, beyond the reach of traditional methods (Rapport, Costanza, and McMichael, 1998). In addition, our approach has the potential to advance the field of blended AI and its applications in environmental research as we aim to create a hierarchy of AI models capable of achieving high accuracy and reliability in classifying bioimages uploaded by citizens without a scientific background. This will potentially lay new foundations, or extend current ones, in addressing broad bioimages classification problems using AI, a field where most work to date has focused on general or specific AI models, but not a hierarchy capable of accurately classifying the flora and fauna of an image through different taxonomic ranks (Solari et al., 2021).

1.1 Motivation and objectives

AMOVALIH's development is fueled by applying Machine Learning (ML) and Deep Learning (DL) methodologies to address real-world problems. Our recent involvement with biodiversity and environmental indicators has underscored the vital role that AI models can play in tackling complex environmental challenges (Ye et al., 2020).

Acknowledging the hurdles associated with data quality, data volume, and the need for interpretability and transparency in AI models, the AMOVALIH approach presents an intriguing challenge. By navigating these challenges, we expect to deepen our understanding of the practical applications of AI, thereby contributing to sustainable development goals (SDGs 2023) related to biodiversity and environmental sustainability.

We are motivated by the belief that our work can contribute significantly to the fields of AI, environmental science, and biodiversity research, while inspiring and empowering citizens to play an active role in protecting our planet's biodiversity.

The primary objectives of this document can be divided into two interconnected facets. First, we aim to generate a variety of AI models and a hierarchy of these models for bioimage classification, leveraging a combination of ML and DL methodologies and algorithms. Second, we provide the design of an API that serves as a conduit for integrating and employing various AI models for bioimage classification. This system will facilitate the emergence of hybrid AI, with the end goal being the integration of this API into a participatory citizen platform. Recognizing the exploratory nature of our approach, we have outlined several secondary objectives that may require adjustments due to the research uncertainties inherent in such pioneering work. These objectives include:

- Identify and employ the most effective preprocessing techniques for bioimages to maximize model performance and accuracy.
- Understand the iterative process between various ML and DL models and ascertain how different methodologies can enhance these models.
- Generate and validate AI models that accurately classify species, delivering valuable insights to citizens and raising their engagement in ecosystem sustainability.
- Prioritize transparency and interpretability in the AI models, ensuring equitable access to information and building trust.
- Develop a hierarchy of models, capable of providing accurate and reliable classification of bioimage data down several taxonomic ranks.
- Design and implement an API that includes both local and external AI models, establishing a diverse and robust classification system.
- Contribute towards achieving the SDGs related to biodiversity and environmental sustainability, particularly Goal 14: Life Below Water 2023 and Goal 15: Life on Land 2023.
- Encourage social responsibility and active citizenship by promoting citizen participation in the data collection process.

Our objectives align with the broader goal of facilitating the development and implementation of hybrid AI, a system that seamlessly blends human intelligence with AI capabilities for decision making. This approach not only leverages the strengths of AI for processing large datasets and making predictions but also taps into human expertise for tasks that require complex judgments and creativity. Through this project, we seek to pave the way for implementing high-quality hybrid AI systems in biodiversity research and beyond.

1.2 Impact on sustainability, ethical-social and diversity

The creation of AI models that accurately classify species and offer valuable insights to citizens can stimulate their engagement in ecosystem sustainability. This work directly contributes to the achievement of the SDGs related to biodiversity and environmental sustainability, specifically Goal 15, which emphasizes the protection, restoration, and promotion of sustainable terrestrial ecosystems, and Goal 14, which focuses on the sustainable use of oceans and marine resources.

A key feature of our approach is the creation of an “API of APIs”, which allows developers to integrate their AI models. This open-source API, designed with transparency, interpretability, and reusability at its core, has a positive ethical and social impact. It aims to foster the democratization of AI by making it easier for developers to contribute their AI models, thereby enhancing access to cutting-edge AI models and contributing to ecosystem diversity (Lins, Pandl, Teigeler, et al., 2021).

Promoting inclusivity and diversity is another core aim of this project. We aim to encourage citizen involvement, regardless of background or scientific expertise, in environmental monitoring initiatives and image data collection. This fosters social responsibility and active citizenship by raising awareness about the importance of preserving the ecosystem and its resources. Moreover, it ensures a diverse range of perspectives and experiences are represented in the data collection process.

1.3 Approach and methods

Implementing a blended AI approach that incorporates ML and DL methodologies involves a diversity of potential approaches and methods. There is a compelling need for extensive exploration of the current state of the art, as well as rigorous experimentation with various combinations of techniques and methodologies.

Our strategy combines human interaction and knowledge extraction with AI processing to attain the desired outcomes. This method enables continuous learning from experts, resulting in AI models with higher potential and accuracy compared to those developed without human interaction (Wu et al., 2021). While this approach may present additional challenges compared to purely AI-based models, the trade-off in enhanced accuracy is highly rewarding (Chou, Cheng, and Y.-W. Wu, 2013).

For human interaction, we plan to leverage the MINKA platform, sourcing feedback on bioimages posted by users and incorporating expert information to corroborate or correct user insights. MINKA users will organize and promote community engagement events, ensuring a steady stream of bioimages and insights essential for improving AI models.

Regarding AI model generation, we have two primary options: constructing a single, comprehensive AI model capable of processing various bioimages and classifying species within them, or building a hierarchy of AI models, with each model playing a specific role in processing and classifying images down to a final layer of specialized models for closely related species (Saha et al., 2021).

Our strategy for constructing the hierarchy of AI models involves using the most recent advancements in the combination of ML and DL methodologies to develop a set of hierarchies that effectively classify bioimages data down several taxonomic ranks (Seventekidis and Giagopoulos, 2021). This is an iterative process, where model configurations are continually tested to meet the project objectives. Our methodological framework encompasses:

- Acquisition of large, accurately labeled bioimage collections.
- Application of preprocessing techniques to bioimages.
- Exploration of the iterative interaction between ML and DL models, incorporating various methodologies
- Development of hierarchical models for accurate bioimage data classification.
- Design of a framework for integrating AI models into participatory platforms.
- Ensuring interpretability, transparency, ethics, and contributions of AI models to the SDGs related to biodiversity and environmental sustainability.

For preprocessing and model construction, we utilize open-source software and libraries like TensorFlow, Keras, Pandas, and Matplotlib. Additionally, Docker containers are employed for generating virtual environments, interoperability, and reusability.

A significant emphasis of our project is the construction of an “API of APIs” to facilitate easy integration of external AI models. This API provides a consistent and standardized interface for external AI models to interact with MINKA observatory, thereby enhancing interoperability, promoting collaboration and enabling Hybrid AI.

The products derived from this effort encompass an array of sophisticated systems and methodologies. Our focus has been to lay the groundwork for future advancements, positioning this work as a blueprint for how similar objectives can be achieved more efficiently given additional time and resources.

Key among our achievements is the construction of a virtual environment with the necessary libraries for deploying ML and DL techniques utilizing GPU power. This environment also includes tools for data visualization, statistical analysis, and a Jupyter Notebook user interface, ensuring that our work is not only reproducible, but also accessible and easily understood.

Further, we have assembled bioimage datasets from MINKA and other sources, and organized them in a generalized structure useful for the creation of various AI models and integration of additional datasets. These images have been meticulously preprocessed, resized, and subjected to data augmentation, especially for species with a smaller number of images.

In the domain of AI model generation, we have devised a broad spectrum of tools and models tailored to different taxonomic ranks. For the kingdom taxonomic rank, we have developed five unique models and one distinct model for each phylum within these kingdoms. These efforts culminated in the creation of a flexible hierarchy of AI models, which is navigated with the help of a JSON configuration file detailing the models' specific hierarchy and characteristics.

Lastly, we have also set up another virtual environment equipped with the necessary libraries to implement the Models API using FastAPI as the primary technology and NoSQL database for persistent and adaptable data storage. This “API of APIs” facilitates the integration of both internal and external AI models. The API boasts endpoints that enable model registration, viewing, modification, and image classification. Moreover, it incorporates endpoints for user feedback, model statistics, and user reputation, thereby fostering the implementation of hybrid AI via continuous model retraining based on users feedback

2. State of the art

We aim to harness and integrate methodologies from two primary AI fields: Machine Learning (ML) and Deep Learning (DL). ML, as the foundational approach for pattern identification and complex problem-solving. However, with the emergence of DL, particularly in image classification, a discernible set of advantages and disadvantages between these two methodologies has surfaced. Comparative analyses indicate that while ML models excel in smaller-scale datasets, DL models outperform in terms of recognition accuracy and are ideally suited for larger-scale datasets (Wang, Fan & Wang, 2021).

Recently, Hybrid AI and Hierarchy AI have become the center of attention within the AI community (Guo et al., 2022), credited to their improved performance and accuracy. These methodologies employ techniques from various AI branches, with an aim to mitigate their individual shortcomings and capitalize on their strengths.

APIs have gained significant importance in the AI ecosystem, enabling the integration and utilization of AI models. They provide a flexible interface where users can integrate their AI models, apply them, and offer valuable feedback (Liang et al., 2023). This fosters the development of Hybrid AI by integrating the feedback loop into the system, promoting continuous model improvement. A robust API facilitates easier and more effective collaboration among AI researchers, developers, and end-users, enhancing model capabilities and accelerating AI evolution.

As MINKA is committed to promoting citizen engagement and aligning with the SDGs, the deployment of ethical AI principles is essential. Such principles ensure that AI deployment is inclusive, respects privacy, promotes transparency, and conforms to legal and societal norms. This chapter aims to delve into the current state of the art in bioimage preprocessing, combined ML and DL methodologies, hierarchy AI, API for AI models integration, blended AI, and ethical AI.

2.1 Bioimage preprocessing

In the realm of bioimage analysis and classification, preprocessing is of paramount importance for achieving a reliable and accurate classification model (Moradmand, Aghamiri, and Ghaderi, 2019). The need to adapt these preprocessing techniques to the specific demands of the problem at hand has been well documented (Bernal, Sánchez, and Vilariño, 2013).

Given our goal to generate a hierarchy of models capable of accurately classifying bioimages down to the phylum level, the quality and size of the image dataset are of crucial significance (Perez and J. Wang, 2017). Our strategy for expanding the dataset involves data augmentation, executed by applying various transformations to the original images. Data augmentation techniques are typically divided into three categories (Xu et al., 2023):

- Model-Free: Traditional transformations such as cropping, zooming, rotating, flipping, and distorting that do not require a pre-trained model.
- Model-Based: Utilize trained AI models.
- Optimization Policy-Based: Ascertain the optimal operations with suitable parameters from a vast parameter space.

Augmentor (Github, 2023) is an open-source library that provides a wealth of functions for applying augmentation techniques documented in the literature (Bloice, Stocker, and Holzinger, 2017). This library can conveniently generate over ten images from one, though a significant tuning component is required: the animals and plants in the images should not undergo excessive deformations, as deformed images across some species may bear similarities.

Image features extraction is another crucial phase in bioimage analysis and classification, particularly when employing ML techniques. Feature extraction techniques aim to reduce data dimensionality and draw out pertinent patterns, noticeably influencing the efficiency and performance of the AI models intended for character recognition (Trier, Jain, and Taxt, 1996). There is an array of feature extraction methods suitable depending on the needs and characteristics of the data (Hall et al., 1971). However, these techniques generally fall into three groups (Kumar & Bhatia, 2014):

- Extraction of statistical distribution features of the points.
- Global transformation and series expansion, such as Fourier, Hadamard, and Rapid transforms.
- Extraction of geometrical and topological features representing both global and local properties of characters, such as strokes.

A plethora of techniques exist for this extraction, and the selection heavily relies on the problem specifications, making it challenging to identify the optimal ones for the dataset at first glance. Nonetheless, it was found that the Bag of Features (BoF) serves as an excellent technique for ML image classification problems, providing a solid starting point (Loussaief & Abdelkrim, 2016).

2.2 Combined Machine Learning and Deep Learning methodologies

The singular application of AI models, while simpler to build, train, and faster in computational speed, are limited in their capabilities (Moazamnia et al., 2019). Current advancements in technology and increased computational power allow for a more robust approach combining multiple AI methodologies, achieving increased accuracy and stability. Despite the requirements of a higher level of knowledge, computational power, and intricate parameter tuning, the integration of multiple AI models has brought about significant enhancements in prediction capabilities and is considered the optimal approach for our bioimage classification problem (Wang & Srinivasan, 2017).

We aim to utilize the ML and DL branches of AI, each offering unique advantages and disadvantages. Therefore, it is essential to explore the most effective methodologies of these branches.

2.2.1 Machine Learning Applied to Bioimages

As an advancement of classical statistical approaches, ML algorithms offer ease of implementation and improved results (Breiman, 2001), hence becoming the core of AI. ML algorithms learn rules from large datasets to identify dominant patterns and make predictions on new data, using different strategies and methodologies based on the characteristics of the dataset and the problem to solve (Loussaief & Abdelkrim, 2016).

The used dataset consists of bioimages, for which the extraction of a large set of features is critical for generating different image classifying ML models. Fortunately, there are many options available for this purpose (Popescu & Sasu, 2014). Once the necessary features are extracted, we can choose models from the most common ML categories:

Supervised Learning

Supervised learning models use labeled data for training and exploration, identifying patterns to map new data inputs into the desired labeled outputs. The primary task of these models is to construct an estimator capable of predicting the label of an object using its extracted features (Nasteski, 2017).

Supervised learning methods fall into the following categories (Ray, 2019):

1. Regression algorithms: Constantly modeling the relationship between variables refined using a measure of error in the predictions made by the model.
2. Instance-based algorithms: Modeling of a decision problem with instances of training data considered important to the model. These algorithms take less time for training but more time for predictions.
3. Decision tree algorithms: Generate a decision tree model that maps the features of a data entry to the output value, with every tree leaf representing a class label and every branch a conjunction of features defining the label class.

4. Bayesian algorithms: Based on Bayesian statistics, these algorithms use Bayes' Theorem to update probabilities based on new data. They create a probabilistic model for classification and regression problems, providing predictions and quantifying the uncertainty associated with them.
5. Artificial Neural Network (ANN) algorithms: While these can be both supervised and unsupervised, we will apply them in the DL context.
6. Deep Learning algorithms: These are complex modern updates of the ANN and can be supervised, unsupervised, and semi-supervised.

The performance of these algorithms depends on the types of data and parameter configurations. Nevertheless, the most used ones with the potential to solve our bioimage classification problem are (Caruana & Niculescu-Mizil, 2006):

Support Vector Machine (SVM): SVM is a traditional ML technique that does not fall into any of the previous categories. It works by finding the best hyperplane that separates data into different classes, handling high-dimensional data well, making it a suitable option for our large-scale bioimage classification problem (Lin et al., 2011).

Logistic regression: Logistic regression is a type of regression algorithm that focuses on explaining a dependent variable from a function with multiple independent variables, i.e., the extracted features (Menard, 2002). It performs well with specific types of data with very descriptive features. However, in image classification problems, its effectiveness is contingent on the appropriateness of the image feature extraction (Li, Bioucas-Dias, & Plaza, 2013).

Naive Bayes: The Naive Bayes algorithm is a type of Bayesian algorithm known for its high accuracy without requiring a training phase. However, it endures substantial computational pressure due to distance computations in the space of local features (Zhu, Jin, Zheng, et al., 2014).

K-Nearest Neighbors (KNN): KNN, a type of instance-based algorithm, decides on the label of an image by searching for the k images in the training set most similar to the image to be classified. It then performs a class-weighted frequency analysis. Its effectiveness in image classification has been demonstrated using local features and interest points extracted with SIFT or SURF to train the algorithm (Amato and Falchi, 2010).

Decision tree: Decision trees, a type of decision tree algorithm, have a top-down branched structure that can generate rules to filter datasets into their appropriate categories/labels effortlessly. Additionally, they can highlight the relative importance of different variables in the system being studied (Yang et al., 2003). Even though they can be unsupervised, it does not apply to our case study.

Unsupervised Learning

Unsupervised learning models are typically used when the data is not labeled, aiming to discover inherent patterns or structure within the data. In contrast to supervised learning models that focus on predicting labels, unsupervised learning models cluster the data into categories based on the data's internal similarities and differences (Barlow, 1989).

Semi-supervised Learning

Semi-supervised learning models are beneficial when a large portion of the data is labeled, and only a small part is unlabeled. These models leverage the unlabeled data to enhance the learning accuracy of the model, while supervised learning only uses the labeled data. In the context of this work, semi-supervised learning models may not be as effective as supervised learning models due to the availability of entirely labeled datasets (Engelen & Hoos, 2020).

2.2.1.4 Reinforcement Learning

While Reinforcement Learning (RL) was not used for our current approach, its significance in the AI field warrants a brief overview. RL models involve an agent learning how to solve a problem through trial-and-error interactions with a dynamic environment. The typical formulation used for RL problems is the Markov Decision Process (MDP) (Barlow, 1989).

In RL, an agent perceives its environment and performs actions to receive a reward or penalty (Kaelbling, Littman, & Moore, 1996). Over time, the agent learns the environment's dynamics, thereby developing a strategy or policy that maximizes the cumulative reward. This learning process differentiates RL from supervised learning, as it does not rely on predefined input/output pairs (Dayan and Niv, 2008).

RL methods fall into three categories:

1. Model-based: The agent learns a model of the environment that predicts the next state and reward based on its actions. The model can be used for planning, allowing the agent to simulate future outcomes and choose actions that lead to the desirable outcomes (Moerland et al., 2023).
2. Model-free: The agent learns directly from experience without building an explicit model of the environment. Examples include Q-learning and SARSA.
3. Actor-critic: A combination of both model-based and model-free learning. The actor learns a policy for selecting actions, while the critic learns to evaluate the actions taken by the actor. This approach can be more effective in complex environments where both planning and learning from experience are important. Examples include A3C and DQN-CA.

Although RL's potential in complex image classification problems has been demonstrated in several cases (Bharati et al., 2022), its application has not been considered yet, primarily due to the lack of literature supporting the use of multiple models as the Q-function in a RL model.

2.2.2 Deep Learning (DL)

DL enables computational models composed of multiple processing layers to learn data representations with various levels of abstraction, thereby uncovering complex structures in extensive datasets. The models utilize the backpropagation algorithm, guiding the adjustment of internal parameters used to compute the representation in each layer from the previous layer (LeCun, Bengio, & Hinton, 2015).

Studies report superior predictive performance of DL techniques applied to image processing tasks compared to traditional classification techniques, especially in complex scenarios. This advantage stems from DL's capacity to directly identify and extract relevant features from a given image dataset (Affonso et al., 2017). Numerous types of DL algorithms exist, but for bioimage classification, Convolutional Neural Networks (CNN) stand out as the optimal choice.

Convolutional Neural Networks (CNN)

CNNs excel in image and video recognition tasks. Their strength lies in learning relevant features directly from raw pixel data while exhibiting robustness to transformations in the input image such as translations, rotations, and scaling (Affonso et al., 2017). CNNs achieve this through a series of convolutional and pooling layers, followed by fully connected layers (Nasr-Esfahani et al., 2019).

The first layer of a CNN is a convolutional layer that applies a set of filters (also known as kernels or feature detectors) to the input image, resulting in a set of feature maps. Each filter is a small matrix of values that convolves with a corresponding region of the input image. The values obtained are then passed through a nonlinear activation function, such as the ReLU (Rectified Linear Unit), to introduce nonlinearity into the network (Talathi & Vartak, 2016).

After several convolutional layers, a pooling layer is introduced to reduce the dimensionality of the feature maps, simultaneously enhancing the network's tolerance to minor variations in the input image. Max pooling is a common type of pooling layer, retaining the maximum value within a small rectangular window of the feature map and discarding the other values (Christlein et al., 2019).

Lastly, one or more fully connected layers map the output of the convolutional and pooling layers to a set of class scores. This process involves flattening the output of the previous layers into a vector, passing it through dense (fully connected) layers, and then applying a softmax function to produce a probability distribution over the output classes (Basha et al., 2020).

During training, the CNN is optimized to minimize a loss function, such as cross-entropy between the predicted class probabilities and the true class labels. This optimization is typically done using backpropagation, computing the gradients of the loss with respect to the network parameters, and using these gradients to update the network weights via an optimization algorithm like stochastic gradient descent (Zhu, Q. et al., 2020).

2.3 Hierarchy AI

Within the context of our approach, Hierarchy AI refers to the organization of multiple layers of models or algorithms, each designed to solve a specific sub-task or problem. The outputs of each layer serve as inputs to the next, culminating in the final output produced by the last layer (Manis & Madhavaram, 2023).

We propose to utilize a hierarchical arrangement of AI models, wherein each model performs a progressively more detailed classification task than the one preceding it. The initial model classifies an image according to the taxonomic rank of a kingdom. Depending on this classification, the image is then processed by the subsequent corresponding model for a deeper level of classification. In an ideal scenario, the final model in this hierarchy would accurately identify the specific species present in the image.

The hierarchical approach in AI is often employed in complex systems involving many different levels of abstraction. In essence, hierarchical AI is a potent method for designing systems capable of learning and reasoning about intricate data in a structured and efficient manner (Huang, Kumar, & Zabih, 1998).

Despite the existence of similar concepts such as multi-layer classification and cascade classification (Song et al., 2019), the literature about constructing a hierarchy AI as defined in our approach is scarce.

2.4 Hybrid AI

Hybrid AI is a relatively new concept, and as such, its interpretation varies within the community. There are two primary interpretations of the terms “blended AI” and “hybrid AI”:

1. Blended AI is seen as the integration of two or more AI technologies to achieve a specific goal, while hybrid AI is the combination of AI models with human knowledge (Dafonte et al., 2020; Ibrahim et al., 2022).
2. Both blended AI and hybrid AI are used interchangeably to denote the amalgamation of two or more AI technologies (Yadav et al., 2023).

Regardless of the specific definitions, both blended and hybrid AI aim to enhance the performance and capabilities of AI systems by combining multiple technologies or approaches. In this document, we will employ the term “blended AI” as the integration of different AI models, while “hybrid AI” refers to the coupling of AI models with human knowledge.

The efficacy of blended AI in enhancing model performance and accuracy has been demonstrated across various fields (Corchado and Aiken, 2002). However, our approach proposes a unique system where an iterative learning process stems from the results of the blended AI. This process works by incorporating human responses into the models’ prediction adjustments. The models consider the human’s feedback, the reliability of the source (the person's reputation), and the model's stats to refine their future responses, thereby achieving the implementation of hybrid AI. Unfortunately, there is scant literature available on this subject.

2.5 Ethical AI

This effort aims to ensure that the public can access, understand, and interpret the data fairly and equitably. We intend to develop comprehensive documentation that emphasizes transparency, interpretability, and reusability to positively influence social ethics (Ahmad et al., 2022). To achieve this, a clear definition and context surrounding ethics and AI are necessary.

AI ethics can be approached from various perspectives, but we focus on a human-centric AI outlook as the emerging overarching value framework of AI ethics (Lepri, Oliver, & Pentland, 2021). There is also a nature-centric AI perspective that considers the natural environment and climate (Kazim & Koshiyama, 2021). This conception of ethics incorporates environmental concerns as centrally as it does human concerns, including humans, animals, and the natural environment. Hence, adopting a nature-centric AI view could lead to a human-centric AI approach.

Given this context, we will follow the Ethics Guidelines for Trustworthy AI 2019, crafted by the High-Level Expert Group on AI appointed by the European Commission, as our guide for implementing an ethical and trustworthy AI (Smuha, 2019). The European Union posits that AI systems should be human-centric, and trustworthiness is prerequisite for people and societies to develop, deploy, and use AI systems. To achieve this trustworthiness, the AI HLEG uses Fundamental Rights as the basis for Trustworthy AI (Smuha, 2019):

- Respect for human dignity.
- Freedom of the individual.
- Respect for democracy, justice, and the rule of law.
- Equality, non-discrimination, and solidarity.
- Citizens' rights.

From these Fundamental Rights spring a list of ethical principles in the context of AI systems: respect for human autonomy, prevention of harm, fairness, and explicability (Hickman & Petrin, 2021)

These principles form the foundation of an Ethical AI following the AI HLEG guidelines (AI, 2019). To integrate this, certain requirements are necessary:

- Human agency and oversight: Fundamental rights, human agency, and human oversight.
- Technical robustness and safety: Resilience to attack, fallback plan, general safety, accuracy, reliability, and reproducibility.
- Privacy and data governance: Respect for privacy, quality and integrity of data, and access to data.
- Transparency: Traceability, explainability, and communication.

- Diversity, non-discrimination, and fairness: Avoidance of unfair bias, accessibility, universal design, and stakeholder participation.
- Societal and environmental wellbeing: Sustainability and environmental friendliness, social impact, society, and democracy.
- Accountability: Auditability, minimization and reporting of negative impacts, trade-offs, and redress.

Different groups of stakeholders should play distinct roles in ensuring these requirements are met (AI, 2019):

- During development, we should implement and apply the requirements to design and develop the processes.
- During deployment, we should ensure that the systems in use and the products and services offered meet the requirements.
- End-users (citizens) should be informed about these requirements and able to request their enforcement.

3. AMOVALIH development

Accurately classifying all known species is a significant challenge due to the constraints of resources and time. To counter this, our approach aims to amplify classification capabilities by developing an API.

The system is designed to empower authorized users to add and select from a broad array of classification models, with an objective to establish a more adaptable and customizable platform for species identification. We intend to accomplish the following through this platform:

1. Acquire and preprocess the requisite quantity of bioimages.
2. Generate AI models for classifying bioimages using ML and DL techniques.
3. Combine AI models using hierarchy AI methodologies.
4. Integrate our AI models into the API, along with enabling the integration of external AI models.
5. Enable hybrid AI by utilizing the users feedback, reputation, and model stats effectively.

A crucial aspect of our targeted system involves implementing a reusable, efficient system to manage image acquisition and AI model generation across multiple datasets. We have established a virtual environment for seamless deployment, thereby simplifying the model creation process, enhancing reproducibility, and ensuring ease of use and compatibility across diverse systems.

In the upcoming sections, we detail our methodology, starting with image acquisition from diverse sources. We will discuss data preprocessing strategies, potential dataset issues, the creation of AI models using advanced ML and DL, and their combination to generate a hierarchy of AI models. We will also cover the integration of these models into the API and the use of hybrid AI to utilize user feedback for model refinement and adaptive learning.

3.1 Image acquisition

To establish a rich, comprehensive dataset, we needed a large quantity of high-quality images representing a vast array of species. This process commenced with the acquisition of research-grade species images from the MINKA dataset. We then generated a CSV file that encapsulated essential data about these images, including associated taxonomic levels, location, and image URLs. This CSV file subsequently played a pivotal role in structuring and organizing our images as it served as the foundation for the organization and structure of our images.

The images were organized in a hierarchical manner, with each image placed within a cluster of folders representing various taxonomic ranks, down to the most specific identified taxon per image. This hierarchical structure has several advantages:

1. Accommodation of varying taxonomic ranks: This structure allows us to use images at different classification levels. For example, an image with information up to the phylum can be used for broader classification models, while an image with full taxonomic information can be employed for refined species-level models.
2. Smooth model training: By segregating images based on their lowest taxon, we improve the performance of our classification models.
3. Integration of additional images: The organized folder structure permits the easy addition of images from various sources, thus enhancing the robustness of our models. New images can be matched to the appropriate taxonomic rank within our structure.
4. Consistent and relevant image inclusion: All images contribute meaningfully to our models, as only images sharing at least one taxonomic rank with the structure from the MINKA dataset are included.

Through the use of Python functions, we download and organize the images, resulting in a collection of 129,823 images covering 4,664 species. To expand our dataset, we added images from other repositories. This enhanced our models' robustness and species coverage, and due to our organized structure, we could seamlessly integrate these new images into our existing dataset for their use in model development and training.

Our dataset has grown substantially and now includes a vast collection of high-quality, research-grade images. Throughout the data collection process, maintaining dataset quality has been a key principle.

Adding images indiscriminately from the internet could risk the quality of our dataset. Lower quality or less accurate images could negatively affect AI models trained on this dataset, leading to less reliable and inaccurate predictions.

Moreover, the principle of diminishing returns comes into play. Each additional image brings a reduced benefit in terms of improving model performance, especially with an already large and diverse dataset like ours. Conversely, the costs and resources needed to process and manage a continually growing dataset keep rising.

Given these considerations, we have decided that our current dataset, balanced across species and providing comprehensive coverage, is sufficient for our needs. We will not source additional images from additional repositories.

3.2 Data preprocessing

Data preprocessing is a vital step in generating robust AI models, particularly when dealing with datasets of variable quality, like in our approach which includes images from MINKA and other sources. To maintain consistency during AI model training, we decided to standardize all images to the same resolution. This process involved both upscaling smaller images to enhance their resolution and downscaling larger ones to decrease their resolution. This standardization allows AI models to extract relevant features.

Before we began preprocessing, we decided to eliminate species with insufficient image numbers.

We set a threshold of 10 images, as generating a minimum of 100 images for AI model training would be impossible, even with image augmentation, if the species had less than 10 images. We used the "remove_species_with_less_than_n_images" function to systematically remove folders with fewer than the set number of images per species. After this process, our dataset was reduced from 524,117 images of 4,664 species to 521,628 images of 4,149 species.

In terms of selecting a common resolution for AI model training, we chose the standard 256x256 resolution (Zhu, J. et al., 2020) for several reasons:

- Preserving image details: A resolution of 256x256 sufficiently captures the details for the model to effectively recognize and interpret image features, ensuring crucial information is not lost during resizing.
- Computational efficiency: This resolution strikes a balance between capturing image detail and computational cost. Higher resolutions might offer more detail, but they would increase processing power requirements and lengthen training times. Conversely, lower resolutions may decrease the computational load but compromise the model's learning effectiveness.
- Compatibility with pre-trained models and popular architectures: Several popular Convolutional Neural Network (CNN) architectures and pre-trained models, such as AlexNet (Alom et al., 2018), VGG (Paymode & Malode, 2022), and ResNet (Ning et al., 2022), are designed or adaptable for 256x256 input images. Resizing images to this standard increases our ability to leverage these architectures and pre-trained models efficiently.

Resizing all images in our dataset to a consistent resolution of 256x256 allows us to balance image detail and computational efficiency, while maintaining compatibility with many widely-used AI models and architectures.

3.2.1 Image resizing

Standardizing our dataset for AI model training required us to resize all images to a 256x256 resolution. We accomplished this using functions found in "images preprocessing.ipynb".

First, we used the "extract_and_save_image_paths" function to pull all image paths from our dataset and store them in a CSV file. Then, the "read_image_paths_from_csv" function read these image paths from the CSV file and placed them in a list for easy access. Lastly, the "resize_image" function resized each image based on its original size. For images that needed an increase in resolution, we used an advanced interpolation method to maintain image quality. Conversely, for images larger than the desired resolution, we applied a simpler interpolation method.

After the resizing process, we successfully resized a total of 519,645 images, significantly reducing our dataset's size from 67.5 GB to 6.59 GB. Although we lost about 0.85% of the dataset due to images with unusual formats that could not be resized, the benefits outweigh the losses. The post-resizing dataset is 90.23% smaller than the original, significantly improving our data processing capabilities and efficiency.

3.2.2 Dataset splitting

Following the image resizing, the next critical step is partitioning our dataset into three subsets: train, test, and validate. This strategic division enables the effective training, validation, and testing of our AI models, helping to avoid overfitting and facilitating accurate model performance evaluation. Consequently, we divided the "resized_images" dataset into three folders: "images_train", "images_test", and "images_validation".

In executing this process, we opted for a stratified sampling technique to maintain the species distribution within each subset. Stratified sampling ensures that the proportion of images per species in the training, testing, and validation sets reflects their original distribution in the entire dataset (Liberty, Lang & Shmakov, 2016). This strategy is critical given the varying number of images per species, as preserving this diversity is vital for our AI models to effectively learn across the complete dataset.

The typical convention for dataset partitioning allocates 70-80% for training, 10-15% for validation, and 10-15% for testing. This distribution provides the AI model with plenty of data for effective learning during the training phase, while also retaining enough data for performance evaluation and validation (Chassagnon, Vakalopolou, & Paragios, 2020). For our dataset, we chose a split of 75% for training, 10% for validation, and 15% for testing. This allocation was thoughtfully decided to optimize the balance between effective learning and precise performance assessment (Kebonye, 2021).

We implemented the required functions to execute the dataset splitting. Here, the "split_and_copy_files_recursive" function generates the requisite folder structures (images_train, images_test, and images_validation), extracts the list of images for each folder, and conducts a stratified sampling to distribute the dataset into training, testing, and validation subsets while preserving the distribution of species.

Regarding folders containing images that do not correspond to a species folder (e.g., the "Animalia" folder), the functions account for images at any taxonomic level. Such images are also considered in the stratified sampling process, ensuring that the distribution of images across the training, testing, and validation sets remains representative of the entire dataset.

After splitting, our dataset is divided into 387,894 training images, 81,758 testing images, and 49,993 validation images. This stratified distribution enhances model training efficiency, mitigates overfitting, and enables precise performance evaluation.

In conclusion, splitting the dataset is a critical step in our pipeline. By dividing our dataset into training, testing, and validation sets, we maintain species distribution, optimize our AI model's learning and generalization, and ensure precise performance evaluation, leading to robust and reliable model outcomes.

3.2.3 Data augmentation

Following the splitting of our dataset into training, testing, and validation sets, the next step involves the data augmentation process. Before initiating data augmentation, it is crucial to conduct dataset splitting to avoid having augmented images derived from the same original image distributed across different sets, which could potentially skew the model's learning process (Fawzi et al., 2016).

Our aim was to address class imbalance within the dataset while ensuring that the split of 75% for training, 10% for validation, and 15% for testing was maintained. To achieve this, we devised a data augmentation strategy that produces a varying number of augmented images for each class, relative to the original number of images within that class (Bryan, 2020). This strategy intends to achieve a balanced dataset by generating more images for underrepresented classes.

The"apply_data_augmentation" function was developed to enable this process. It goes through the "images_train", "images_test", and "images_validation" folders applying data augmentation on each species. The target image count per species was defined as 82 for the training folder, 16 for the testing one, and 10 for the validation one, which determined the needed augmentations. This approach balanced the dataset while keeping the overall proportions.

Using the "albumentations" library (Albumentations, 2023), we employed techniques such as rotation, flipping, scaling, shearing, and brightness and contrast adjustments, carefully chosen to ensure congruence with original images and introduce realistic variations. These techniques are applied with parameters selected randomly within a defined range.

After augmentation, the image distribution was 616,019 for training, 99,336 for testing, and 60,627 for validation. To assess the impact of data augmentation on the class imbalance issue, we calculated and compared the standard deviation of image counts for each species in the training, testing, and validation sets, before and after data augmentation (Figure 1).

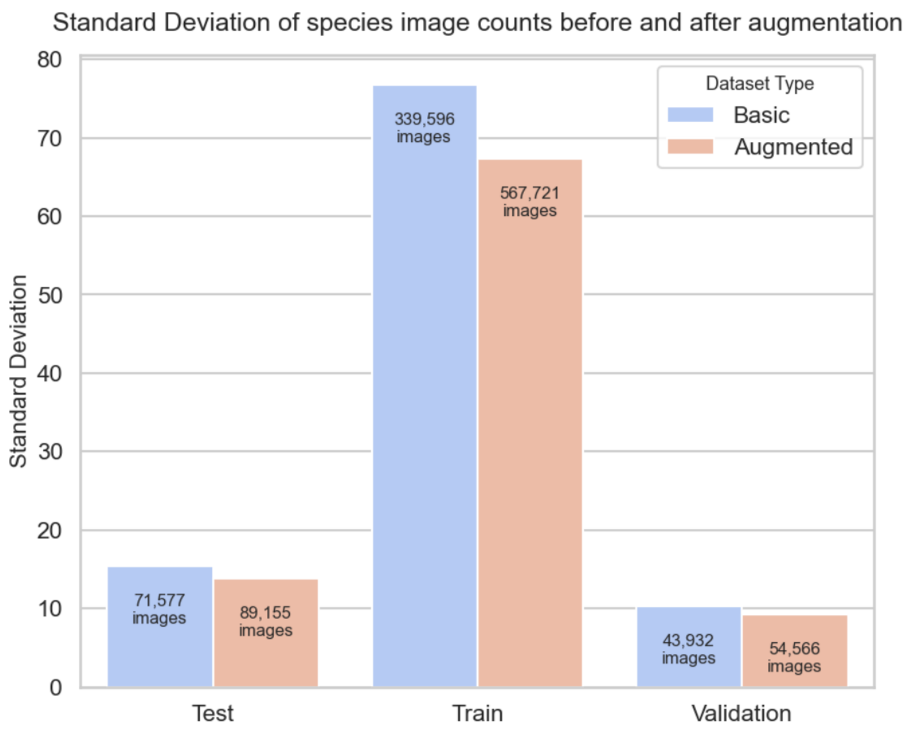


Figure1. Effect of data augmentation on class imbalance: Comparison of Standard Deviation in image counts per species.

We can observe a decrease in standard deviation while the number of images increases indicating that the data augmentation process has effectively reduced the class imbalance, allowing our AI model access to more balanced and diverse training data, thus enhancing its overall performance.

3.4 Dataset derived problems

Addressing the challenges of our sizable dataset is crucial for constructing an efficient model. Initially, we used a Convolutional Neural Network (CNN) model to classify images into five major taxonomic ranks. Despite its simplicity, it required substantial training time on high-performance GPUs. After reaching 27% accuracy, the improvement rate significantly slowed, implying over a day of continuous training would be required, signaling inefficiency and overfitting potential (Bilbao & Bilbao, 2017).

To address this, we modified the “load_dataset” function to select a smaller, random subset of images for training, validation, and testing. This adjustment expedited training while preserving data diversity and representation (Ying, 2019). Consequently, we selected 75,000, 10,000, and

15,000 random images for training, validation, and testing, respectively. The trained model was saved as a structured data file.

Despite the improvements, this method is not the most efficient solution. Thus, we further explored other techniques to optimize our model's performance and reduce training time.

3.2.4 Feature extraction and selection

Mitigating dataset complexity via feature extraction and selection can potentially enhance a model's performance and reduce training time (Cai et al., 2018). We used pre-trained CNN models like ResNet50, VGG16, and InceptionV3, which capture relevant features from images effectively, having been trained on extensive datasets like ImageNet (Vijayan & Sherly, 2019).

We selected the ResNet50 model for feature extraction due to its balance between complexity and accuracy and its resistance to the vanishing gradient problem thanks to its skip connections and residual learning technique (Kumar, Kakarla, Isunuri, et al., 2021). This approach allows us to train models on reduced-dimensionality representations of our images, improving efficiency, especially when dealing with extensive datasets and limited resources.

Moreover, we will train DL models directly on images to potentially increase classification accuracy, feature extraction and selection serve as complementary methods that enable efficient model development and testing (Hira & Gillies, 2015).

3.2.5 Quality of the images

Even though we used research-grade images, some species still include low-quality images, either poor in quality or unrelated to their respective taxonomic ranks. With over 600,000 images, manually cleaning the dataset is impractical. A potential solution is creating various AI models using different approaches and technologies, forming a strategy resilient to the variable image quality (Murtaza, Shuib, Abdul Wahab, et al., 2020).

Ensemble learning techniques might prove useful in this regard, combining multiple models to enhance performance and robustness (Sagi & Rokach, 2018). Different models may excel in different areas, and ensemble methods can leverage these strengths (Xiao et al., 2018). This approach also adds resilience against potential data anomalies.

In conclusion, the primary dataset's challenges are image quality and volume. This involves combining advanced image processing techniques, diverse AI modeling strategies, and ensemble methods. We anticipate this approach to result in a robust and efficient system for image classification despite dataset quality variance. The proposed strategies should improve our classification performance while ensuring computational efficiency.

3.3 Model development

Developing our AI models requires a methodical blend of AI techniques, particularly ML and DL. We aimed for a thorough hierarchy of models that could effectively classify species from bioimages. This methodology also includes a participatory platform and reputation system to enhance model performance and foster a collaborative AI environment.

This section details key processes in our model development. We begin with the creation of the Utils library, a cornerstone of our modeling process. This is followed by feature extraction from bioimages. Next, we discuss the design and integration of our kingdom and phylum models. Lastly, we examine how these distinct models can form a hierarchy, decoding the intricacies of species classification.

3.3.1 Utils

The essential utility functions include dataset generation and normalization, model checkpoints, visualizations, and loading and saving of datasets and models. In this section, we delve into the details of these utilities and their role in our model development process.

Dataset Generation and Normalization

The cornerstone of our model development is the dataset generation using the "load_image_list" function. This function compiles a list of image paths and their associated classes from a specified directory. If a sample size is given, it randomly selects a balanced number of images from each class, preventing class-specific biases and enhancing model generalization.

Following generation, the "process_images" function is employed for image processing and normalization. Each image is transformed into a numpy array, and the pixel values are normalized. This standardization aids the optimizer in converging faster and prevents it from getting stuck in local minima or prolonged weight optimizations (Sun et al., 2020).

Save and Load Datasets

Post generation and normalization, "save_dataset" and "load_dataset" functions are employed to handle the datasets. The "save dataset" function saves the X and y arrays for train, test, and validation datasets as ".npy" files in a specified folder, allowing for easy future access without redoing the initial processes.

On the other hand, the "load_dataset" function retrieves these saved datasets, saving computational resources and time when reusing the same datasets. Additionally, this function logs essential information like file paths and dimensions of the loaded datasets, supporting debugging, understanding dataset status at various stages, and maintaining traceability. Together, these functions enhance productivity and smooth the AI model development process.

Model checkpoints and custom stops

Training complex AI models often encounters disruptions like system crashes, long execution times, or overfitting (Pham et al., 2020). To address these, we have created two custom callback classes: StopOnAccuracy and CustomModelCheckpoint.

StopOnAccuracy helps prevent overfitting by terminating training once a settled accuracy level is reached. This stops the model from becoming overly specialized to training data, aiding its generalization capability.

In contrast, CustomModelCheckpoint periodically saves the model state when notable accuracy improvement is observed. This safeguard is crucial given the time-consuming nature of model training. By saving at regular intervals, we ensure progress is not lost during disruptions, providing a contingency plan.

Together, these classes optimize training, ensuring high accuracy, combating overfitting, and preserving progress, leading to a more efficient and reliable model development.

Simple and Complex CNN model

We streamlined simple CNN model creation with the “create_simple_cnn” function. It uses image dimensions and the number of dataset classes to build a CNN model, designed for robustness and speed. The model includes convolutional layers for local feature capture, max pooling layers for dimensionality reduction, a dropout layer to mitigate overfitting, and a final fully connected (Dense) layer with softmax activation for class probability distribution (Christlein et al., 2019).

Hyperparameters like the number of filters, kernel size, activation function, and dropout rate are selected based on best practices and empirical observations. For example, the ReLU activation function is favored in CNNs for its efficiency in handling the vanishing gradient problem (Talathi & Vartak, 2016).

For more complex tasks, advanced CNN models can be created using “create_complex_cnn”, “create_more_complex_cnn”, and “create_efficientnet_model functions”. These handle intricate image patterns and higher-dimensional data.

The “train_cnn_model” function is key to the model training process. It takes the model, a model name, training and testing datasets, and a set of hyperparameters to start training. This suite of functions allows creating and training both simple and complex CNN models, catering to different complexity levels.

Model visualization

Model visualization and analysis are vital in understanding model performance and learning patterns (Krstinic et al., 2020). For this, we have developed four functions:

plot_confusion_matrix", "plot_confusion_matrix_with_error_rate", "plot_training_history", and "plot_sample_images_per_class".

The function "plot_confusion_matrix" takes the model, test data, and class names to generate a confusion matrix heatmap. This allows us to gauge the model's performance on test data and highlight any class the model struggles with (Krstinic et al., 2020).

The "plot_confusion_matrix_with_error_rate" function, like "plot_confusion_matrix", displays a confusion matrix but also incorporates error rate. This function helps understand the model's accuracy in terms of percentage, showcasing how frequently the model makes incorrect predictions. This provides an enhanced view of model performance for each class.

The "plot_training_history" function tracks and visualizes the model's learning process over epochs by plotting accuracy and loss values for both training and validation datasets. Observing the progression of these metrics offers crucial insights into how the model learns and adjusts its internal parameters over the training process (Shekar, Revathy, & Goud, 2020). For instance, if the training loss consistently decreases while the validation loss begins to increase, it may be an indication of overfitting. Conversely, if the training and validation losses decrease while the accuracy remains low, it could signal underfitting. Thus, "plot_training_history" assists in identifying such scenarios, enabling model refinement.

Finally, the "plot_sample_images_per_class" function displays sample images from the training, validation, and testing datasets for each class, providing a qualitative view of the model's performance.

These visualization and analysis functions facilitate a comprehensive understanding of the model's performance, behavior, and learning trajectory, aiding their fine-tuning and optimization.

3.3.2 Feature extraction

Feature extraction is crucial in ML and DL tasks, it aims to condense image data into a lower-dimensional form while retaining significant and distinguishable information (Elharrouss et al., 2022).

Initially, we used various techniques like calculating histogram descriptors, texture analysis using the Gray-Level Co-occurrence Matrix (GLCM), and using pre trained CNNs like VGG16 for feature extraction. However, the huge data output from this approach posed significant computational challenges.

Even using Principal Component Analysis (PCA) for dimensionality reduction could not alleviate the computational burden. With processing times estimated to span days and a risk of system failure due to excessive memory usage, we had to devise an alternative strategy that catered to our computational limitations while ensuring consistent feature extraction across all datasets and new images.

Traditional image processing techniques

At this stage of processing, we introduced three primary functions.

The “color_space_transform” function enables the transformation of the image's color space, allowing us to choose a color space that best highlights the features of interest in our images, such as HSV or LAB (Bora, Gupta, & Khan, 2015).

The “texture_analysis” function provides means to extract texture-based features. Given the countless textures in biological images, this function offers a robust mechanism to capture these intricate details. It incorporates two texture analysis methods: GLCM” and LBP (Ding, 2017).

Lastly, the “histogram_descriptors” function assists in extracting color-based features by calculating histogram descriptors. Histograms provide a powerful representation of an image’s color distribution (Desai, Pujari, Akhila, et al., 2021).

Feature extraction using pre-trained CNNs

We use the “feature_extraction_cnn” function to unlock intricate, high-level feature patterns within the images leveraging pre-trained CNNs. CNNs are renowned for their efficiency in discerning complex patterns in image data, making them invaluable in our context (Bailer et al., 2018).

Dimensionality reduction and adjusted feature extraction strategy

Given the high-dimensionality of the feature space, efficient management and processing become challenging. For this, we use the “dimensionality_reduction” function, implementing Principal Component Analysis (PCA) to condense our feature set (Thomaz & Giraldi, 2010).

However, we adjust our feature extraction strategy to meet our specific needs and constraints. We fix the number of components to a constant value of 20, ensuring a consistent number of features for each image, making it a flexible approach applicable to new images with similar characteristics. While this method provides stability and consistency, we must consider the potential drawback of retaining less important dimensions or unintentionally excluding significant ones.

Feature selection and evaluation

We refine the feature set further with the “feature_selection” function, which performs feature selection using Recursive Feature Elimination (RFE), an iterative method that determines the best or the worst performing feature at each iteration (Yan & Zhang, 2015).

This approach identifies the most significant features contributing to the model’s performance, thereby allowing us to focus our computational resources effectively. The function provides us with control over the complexity and interpretability of our final model by letting us specify the number of features we want to retain.

Feature extraction and saving in batches

To address concerns regarding computational efficiency and memory management, our feature extraction strategy employs a batch processing approach. The “extract_features_batch_pca” function lies at the heart of this strategy, enabling the loading and processing of images in batches.

The function applies each pre-trained model to an image batch and concatenates the extracted features. If a path to an existing PCA model is provided, the function uses this model to transform the features; otherwise, it fits a new PCA model on the features and saves it, aligning with our modified feature extraction strategy.

For feature extraction from a single image, we use the “extract_features_single_image” function, mainly useful for classifying new images introduced into the system.

The combination of PCA and batch processing creates a computationally efficient method that preserves essential features, laying a robust foundation for subsequent ML tasks.

Application to dataset and new images

We applied our feature extraction strategy to the “kingdom_100000” dataset (subset of 68,662 processed images), using the “load_datasets” function to smoothly load our training, validation, and test sets.

We leveraged the power of three pre-trained models: InceptionV3, VGG16, and ResNet50. For each model, we defined the corresponding preprocessing functions, facilitating the extraction of a diverse set of features from the images. Our choice of these models was driven by their demonstrated robust performance in various image classification tasks (Stefenon, Yow, Nied, et al., 2022).

The “extract_features_batch_pca” function was critical in extracting features from our training, test, and validation sets. Each set of extracted features was saved in separate CSV files, providing a consistent and reusable data representation for our AI models.

The size of our dataset experienced a significant reduction post feature extraction. From an initial size of 39,427 MB, we ended with a considerably smaller one of 15.13 MB, an astounding 99,96% reduction in size (Figure 2).

Our feature extraction strategy strikes an optimal balance between computational efficiency and feature preservation. By employing PCA with a fixed number of components, batch processing, and leveraging pre-trained models, we have successfully extracted meaningful features from a large image dataset without overloading system memory. This strategy sets a robust foundation for subsequent steps in our pipeline, including the training of ML models and the classification of new images.

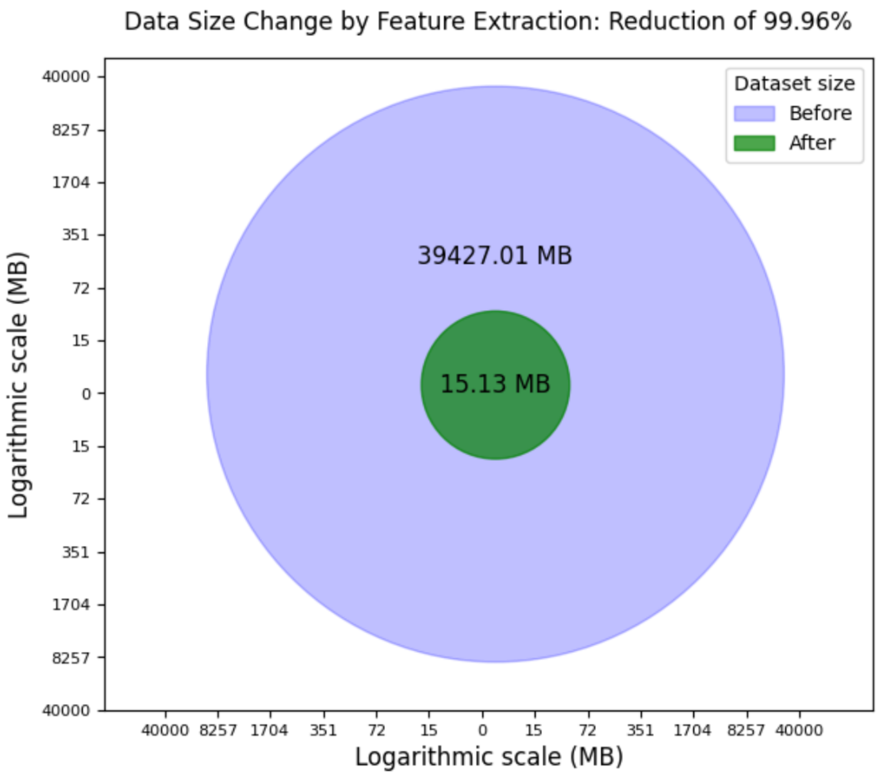


Figure 2. Dramatic reduction in data size from the original “kingdom 100000” dataset to the feature-extracted dataset.

3.3.3 Kingdom taxon

This section focuses on outlining our approach to classify the high-ranking taxa, leveraging a comprehensive strategy that incorporates both DL and ML methodologies.

Our roadmap includes the design, training, and refinement of various AI models of different complexities and diverse feature combinations, with the goal of optimizing the precision of bioimage classifications. Significantly, our process integrates both raw features of the images, and predictions generated by AI models. This integration creates a resilient framework that capitalizes on the strengths of multiple models for superior results.

The Kingdom taxon consists of five categories: “Animalia”, “Plantae”, “Fungi”, “Chromista”, and “Protozoa”. We describe our effort to engineer, instruct, and assess a selection of models responsible for reliably categorizing images into these discrete groups. Each model undergoes a singular cycle of development, training, and evaluation, with the outcomes from each stage guiding the configuration and implementation of subsequent models.

For this classification task, we used the “kingdom 100000” dataset, which includes a curated collection of 68,662 processed images. The dataset is partitioned as follows:

- X_train: 4-dimensional structure with dimensions (51792, 224, 224, 3).
- y_train: 2-dimensional structure with dimensions (51792, 5).
- X_test: 4-dimensional structure with dimensions (6656, 224, 224, 3).
- y_test: 2-dimensional structure with dimensions (6656, 5).
- X_valid: 4-dimensional structure with dimensions (10214, 224, 224, 3).
- y_valid: 2-dimensional structure with dimensions (10214, 5).

Upon exploring the dataset and randomly sampling an image from each class and dataset, we observed that despite the high-quality, research-grade images, certain images could potentially challenge the classification process. As shown in Figure 3, images from categories like “Fungi”, “Chromista”, and “Protozoa” may seem indistinguishable from one another and occasionally resemble those from “Plantae”. The “Protozoa” category, characterized by a wide variety of images ranging from plain text documents to images containing species from the “Plantae” and “Fungi” taxonomic ranks, might confuse the model. Similarly, the “Animalia” category could pose challenges due to its collection of nearly-black images.

While our objective is to classify as many images as accurately as possible, we acknowledge that some images may not be classified correctly due to inherent ambiguity. This inevitable challenge in the realm of model development is considered in our approach, setting realistic expectations and ensuring a robust methodology.

All Kingdom taxonomic ranks

In this section, we present our initial strategy using a Convolutional Neural Network (CNN) to classify images across five kingdoms. Our method involved an extensive exploration of various CNN architectures, activation functions, and hyperparameters, aiming to exploit the rich image content most effectively. We iteratively refined our model structure, balancing complexity and performance while avoiding overfitting.

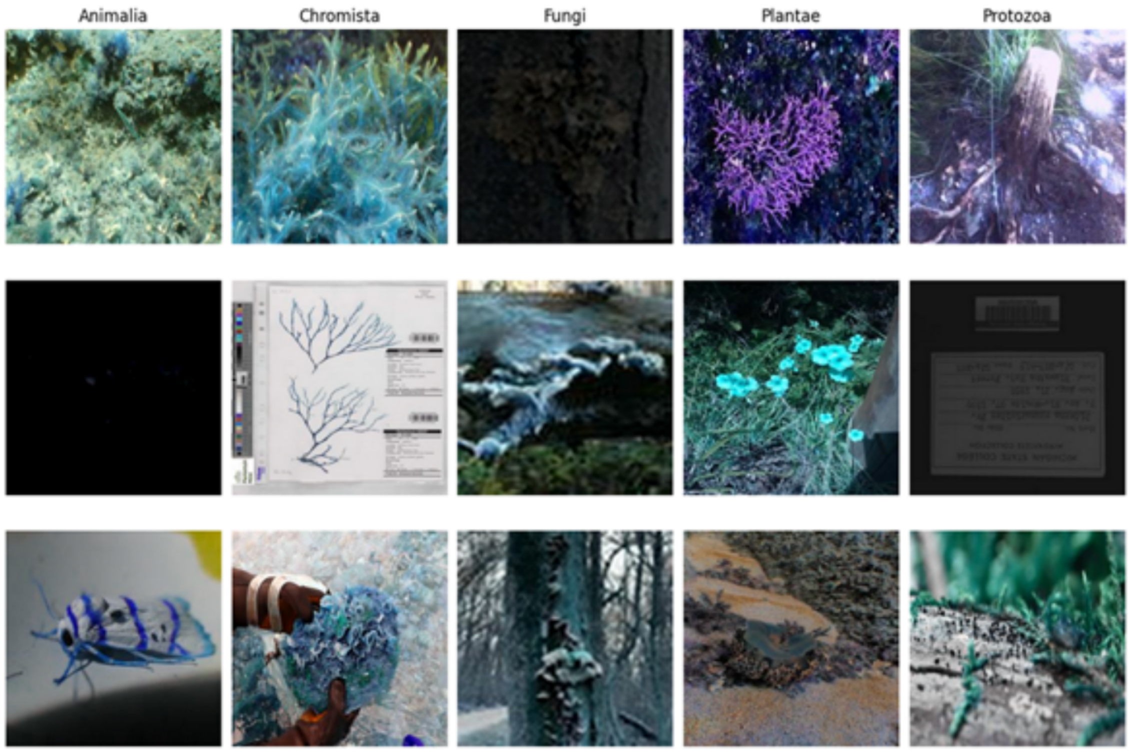


Figure 3. Grid displaying a representative image from each class across the train, test, and validation datasets.

Our selected hyperparameters were a batch size of 32 and 20 training epochs, balancing computational efficiency and performance. We chose a learning rate of 0.0001 to enable a controlled gradient descent during training. The CNN model extracted and learned the most informative image features for classification, and we assessed its effectiveness through total accuracy and a confusion matrix using the test data (Figure 4).

The model achieved an overall accuracy of 59.45%, a reasonable initial performance given the task's complexity. The confusion matrix further illustrated this with challenges in accurately classifying "Chromista" and "Protozoa" images affecting the model's performance.

In particular, "Chromista" and "Protozoa" images were correctly classified about 30% and 10% of the time, respectively, often being misclassified as "Fungi". Given these insights, we decided to generate three additional models to further navigate the intricacies of our dataset and enhance classification performance.

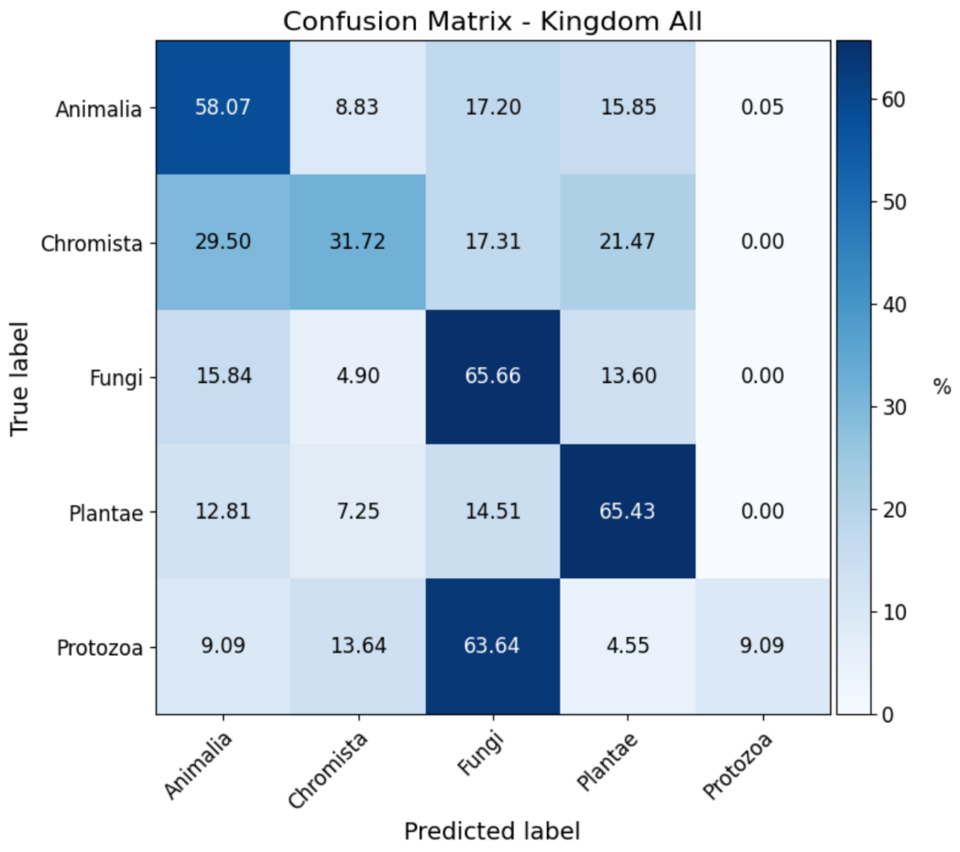


Figure 4. Confusion matrix showing the classification results of the Kingdom All model.

Animalia, Plantae, Fungi - Protozoa, Chromista

Building on the experience gained from the initial CNN model, we identified an opportunity to enhance performance by creating a distinction between the “Animalia”, “Plantae”, and “Fungi” (APF) categories and the “Protozoa” and “Chromista” (PC) categories. This distinction aimed to balance precise classification and the challenges arising from image quality variations, particularly within the “Protozoa” and “Chromista” categories.

With slightly adjusted hyperparameters, we utilized the same CNN model as before, achieving promising accuracy rates between 70% and 80% with minimal overfitting. These outcomes indicated that our model was learning effectively from the training data, maintaining its predictive performance on unseen data.

Despite these positive results, we attempted to enhance our model’s performance further, given its potential as a robust basis for developing hierarchical AI models. We tried to incorporate pre-trained models, such as EfficientNet (Tan & Le, 2019), but these attempts did not provide the anticipated improvements.

Without overfitting as a concern, we decided to construct a more complex CNN model. Using the “create_complex_cnn” function, we constructed a deeper network with additional convolutional and pooling layers, potentially improving classification performance by extracting more intricate image features.

Following training, we evaluated the model’s performance using total accuracy and a confusion matrix based on the test data (Figure 5).

The model achieved an accuracy of 89.53%, a significant figure considering the task’s complexity and some image quality. The confusion matrix underscored the model’s near-perfect performance in classifying the APF bioimages, with 98.49% accuracy. However, it struggled with the PC bioimages, achieving only 18.28% accuracy. Even so, our model is adept at identifying images from these categories, thereby providing valuable insights. As next steps we plan to develop two more specialized models: one for APF images and another for PC images.

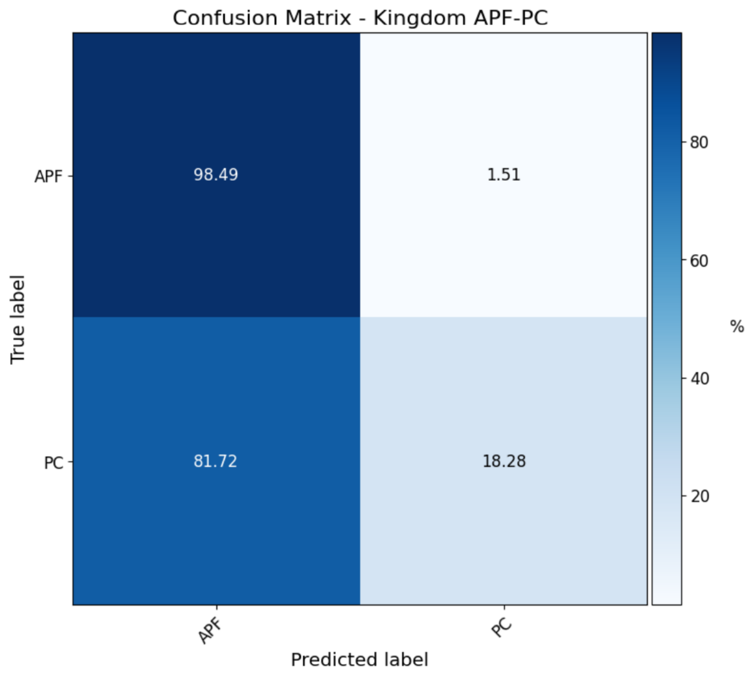


Figure 5. Confusion matrix showing the classification results of the Kingdom APF-PC model.

Animalia - Plantae - Fungi

In response to the initial models’ performance, we focused our efforts on the “Animalia”, “Plantae”, and “Fungi” (APF) categories to further enhance classification accuracy. This specialized approach lessened task complexity and enabled a concentrated exploration of the unique traits within the APF categories, thereby improving the model’s predictive ability.

Our efforts to develop this APF model encompassed an extensive exploration of network architectures, hyperparameters, and training techniques. Despite the EfficientNet model's substantial performance improvement, the complex CNN model introduced earlier proved most effective and accurate. More complex CNN models struggled to exceed 60% accuracy, with slower learning rates.

The chosen model's superior performance was attributed to its additional convolutional and pooling layers, which extracted more detailed and high-level image features, a key factor for boosting classification performance.

Following training, we evaluated the Kingdom APF model's performance using its total accuracy and a confusion matrix derived from the test data (Figure 6).

The Kingdom APF model achieved an accuracy of 73.27%. This is a substantial accomplishment given the task's complexity and the dataset's varying image quality. Closer inspection of the confusion matrix reveals varied performance across categories: around 80% accuracy for "Fungi" and "Plantae" and about 55% for "Animalia". Although the initial model performed better on "Animalia" images, the Kingdom APF model provides the highest overall accuracy.

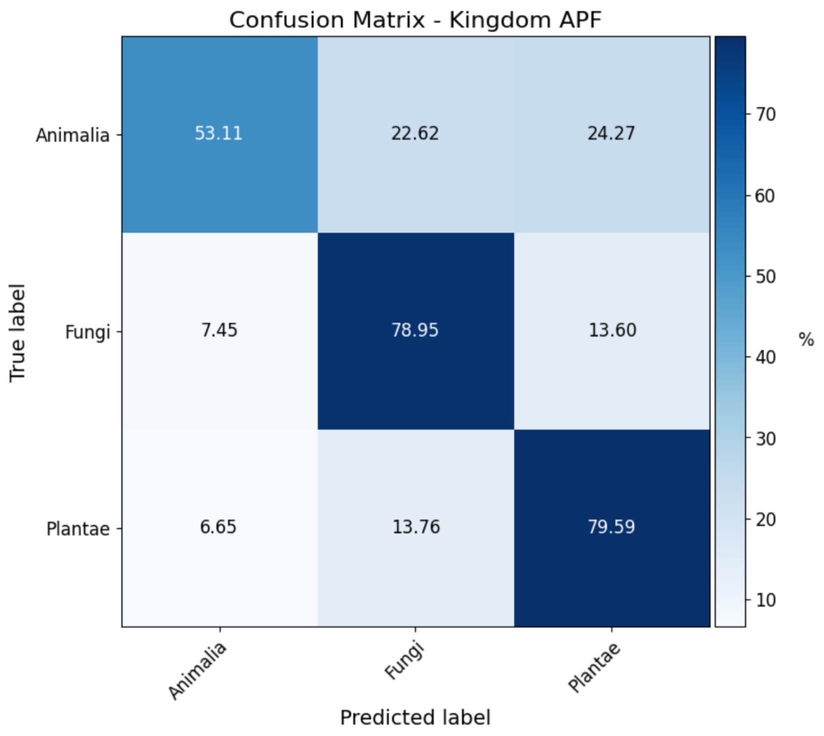


Figure 6. Confusion matrix showing the classification results of the Kingdom APF model

Protozoa - Chromista

Addressing the initial models’ struggles with “Protozoa” and “Chromista” categories, we created a tailored Kingdom PC model, sharpening its focus on their unique traits to enhance classification accuracy.

A key challenge was the discordant number of training images for these categories, which led to over-prediction of the “Chromista” class. We resolved this by balancing our training set to contain equal numbers of “Protozoa” and “Chromista” images, facilitating a more unbiased learning environment and improved precision.

With a reduced dataset after balancing, we increased the number of training epochs. We explored various model architectures and hyperparameters, including pre-trained models like EfficientNet B0 and B3, and custom CNN models with different parameters. Finally, the complex CNN model introduced earlier proved most effective and accurate.

Upon completing the training phase, we assessed the PC model’s performance by calculating its total accuracy and generating a confusion matrix using test data (Figure 7).

The Kingdom PC model achieved an overall accuracy of 92.34%. Due to balanced training data, the model had equal learning opportunities across both classes. The slightly lower “Protozoa” accuracy could be due to inherent complexity or image quality variations. However, the marked improvement from our initial models underscores our targeted approach’s success.

With these four models, each offering unique insights into different categories, we are now prepared for the model stacking phase. Our goal is to leverage each model’s strengths, creating an ensemble for more robust and precise bioimage classification.

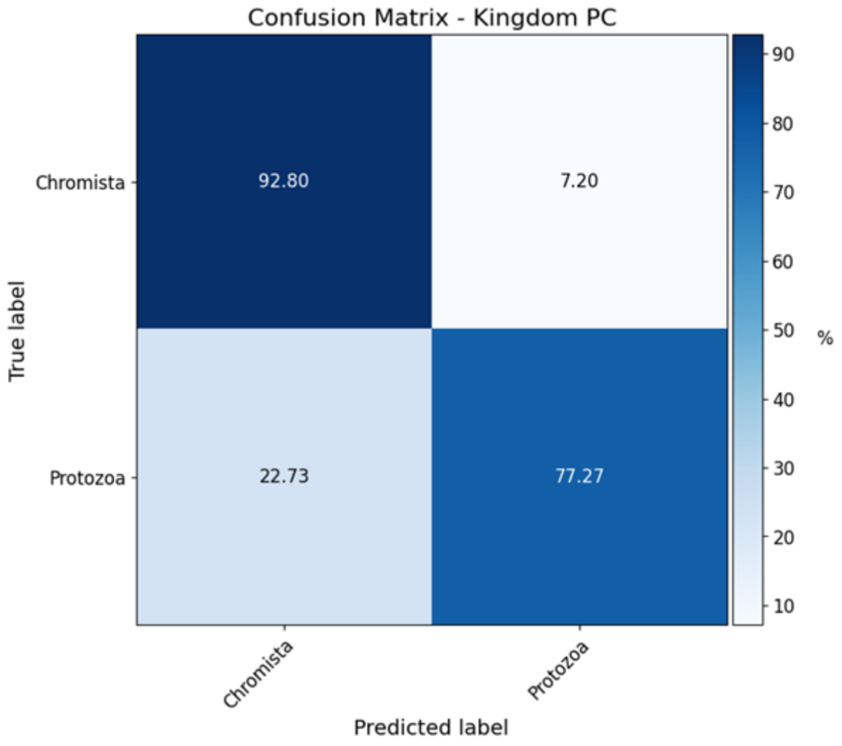


Figure 7. Confusion matrix showing the classification results of the Kingdom PC model.

Model Stacking - Ensemble Learning

To boost our predictive capabilities, we employed a strategy known as model stacking or ensemble learning (Cui et al., 2021). This method combines the predictions of our four models, feeding them as input features to a higher-level model, hence enhancing our dataset with combined insights from the lower-level models.

The function “get_model_predictions” was employed to extract predictions from a set of preprocessed bioimages using a variety of models. This enriched our feature dataset with 12 additional columns, each representing predictions from one of the four models.

However, exclusively applying ML models to the base models’ predictions did not yield satisfactory outcomes. The confusion matrices bore a striking resemblance to the first model’s results, with slightly lower accuracy, particularly for the “Animalia” class. Consequently, we archived these 12 prediction columns in CSV files and later integrated them with the features of CSV, generating comprehensive files with 33 columns for training, validation, and testing datasets.

Upon verifying the integrity of these combined datasets, we trained various ML models, including Support Vector Machines (SVM), Random Forest, and Gradient Boosting. Despite extensive testing and fine-tuning, none of these models surpassed 60% accuracy. The Receiver

Operating Characteristic (ROC) curve for the SVM model is presented (Figure 8) to illustrate its performance across different classification thresholds. We display the ROC curve of the SVM as it provided comparable results to the other models but required fewer computational resources.

The ROC curve is a graphical representation that illustrates the performance of a binary classifier system as its discrimination threshold is varied. It plots the true positive rate (TPR) against the false positive rate (FPR). The area under the curve (AUC) provides a collective measure of performance across all potential classification thresholds where an AUC closer to 1 signifies superior model performance (Carter et al., 2016).

The AUC for “Animalia” is 0.678, implying that the SVM model has a 67.8% chance of ranking a randomly chosen “Animalia” instance higher than a randomly chosen non-“Animalia” instance. Similarly, the AUCs for “Chromista”, “Fungi”, “Plantae”, and “Protozoa” are 0.66, 0.74, 0.77, and 0.56, respectively.

The underwhelming performance of the ensemble models can be attributed to the high correlation among the prediction errors of the base models, thus constraining the room for improvement.

Moreover, the base models’ predictions were impacted by factors such as unbalanced data, intricate and overlapping morphological characteristics, and limitations of the features and models used. Unless improvements are made to the base models, the ensemble models’ performance is likely to remain poor.

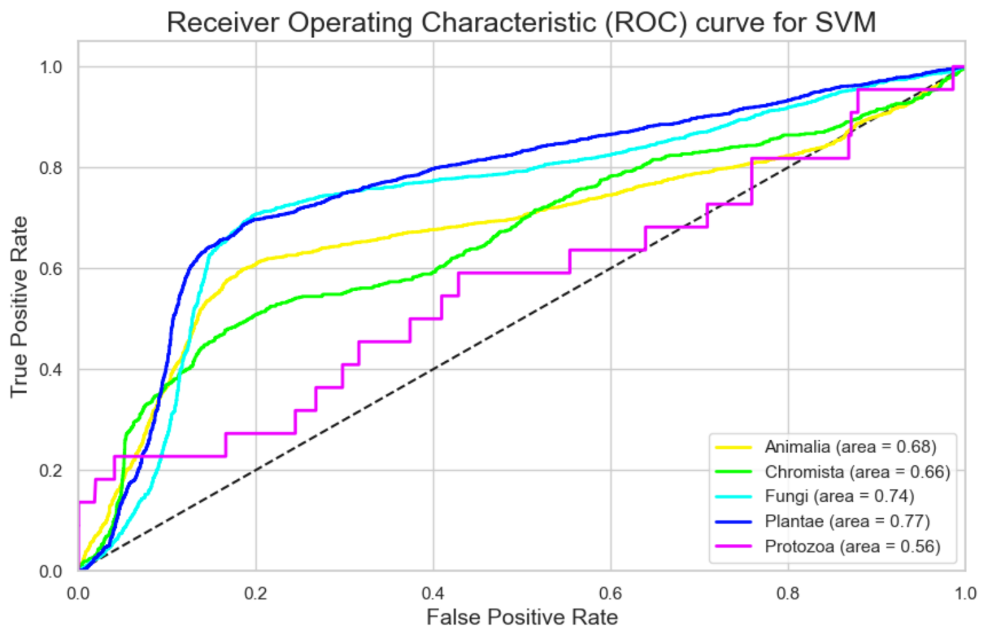


Figure 8. ROC curves illustrating the performance of the SVM model in classifying the classes “Animalia”, “Chromista”, “Fungi”, “Plantae”, and “Protozoa”.

3.3.4 Phylum taxon

Progressing further into the intricacies of biological taxonomy, we shift our focus from the broad scope of kingdoms to the more granular differentiation of phyla. This level offers a detailed perspective on organisms, sorting them based on common lineage and unique characteristics. This section explores the creation and performance of models aimed at classifying organisms into their corresponding phyla.

Our aim, considering our resource and time constraints, is not to supersede current AI models. Instead, our focus is to demonstrate a methodology for constructing a hierarchical AI system. We strive to build proficient models for each kingdom, thereby creating a model suite capable of classifying from kingdom down to phylum. This sets the groundwork for a hierarchical AI system for bioimage classification, linking the outputs of kingdom models to their respective phylum counterparts.

Our strategy at this stage deviates from the one employed at the kingdom level. Rather than creating multiple models for the different phyla, as done previously, we have adopted a different approach: a single comprehensive model for each kingdom. This method, while potentially sacrificing a degree of phylum-specific accuracy, equips us with the essential models to construct a hierarchical AI system.

Animalia

We commenced by evenly selecting 100,000 images from the 12 available phyla within the “Animalia” kingdom. Three of these phyla: “Ctenophora”, “Nemertea”, and “Spuncula”, had fewer images, but still provided ample data for effective model training and testing.

Upon examining a subset of images, we noticed substantial similarities across phyla. Image quality varied, and common elements, such as hands, abundant vegetation, and multiple animals, were frequently observed. These factors pose a challenging classification task.

We dismissed the use of EfficientNet models due to their prior underperformance with similar images. We instead chose a pragmatic approach and experimented with the three CNN models of different complexity levels. The model of moderate complexity outperformed the others, reaching an overall accuracy of 34.1%. While seemingly low, considering the context of 12 classes, varying image quality, and given our illustrative purposes, this performance was deemed satisfactory.

Plantae

For the “Plantae” kingdom, we started with a balanced selection of 100,000 images from the 9 available phyla. Three phyla: “Marchantiophyta”, “Equisetophyta”, and “Pteridophyta”, had fewer images. The images showcased extensive variation within phyla, including low-quality images, dense vegetation, and images of plants on monitors.

Once again, the medium complexity CNN model delivered the best results, despite some signs of overfitting. The model achieved a commendable 73.19% accuracy, although it struggled with phyla represented by fewer images. Notably, the “Marchantiophyta” phylum was frequently misclassified, suggesting the need for a specialized model. However, in line with our project’s illustrative nature, we refrained from creating phylum-specific models.

While the accuracy is commendable, it is worth mentioning that there are already highly optimized models for plant classification, such as those under the Pl@ntNet 2023 initiative. Our primary objective, however, still remains to showcase an effective methodology, rather than surpassing these models.

Fungi

Our analysis of the “Fungi” kingdom began with 100,000 images evenly drawn from the four phyla. We faced a significant class imbalance, with “Ascomycota” and “Basidiomycota” better represented than “Microspora” and “Zygomycota”. The latter two lacked sufficient data for testing and validation, leading us to focus on “Ascomycota” and “Basidiomycota”.

Using various CNN models, the medium complexity model emerged as the top performer, showing improved training and validation set accuracy. Despite the “Fungi” kingdom’s inherent challenges, our final model achieved a notable 84.59% accuracy, validating our targeted approach.

Chromista

The “Chromista” kingdom dataset only comprised the “Ochrophyta” phylum. To enhance our taxonomy, we aimed to classify images into two classes within this phylum: “Coscinodiscophyceae” and “Phaeophyceae”, using around 6,000 images.

Despite some similarities, most images showed distinguishable class characteristics. As in previous instances, the medium complexity CNN model performed best, showing no overfitting and achieving a remarkable 94.38% accuracy. This underscores the potential of DL in classifying similar classes within a specific phylum, even with fewer images.

Protozoa

In the “Protozoa” kingdom, our data only included the “Mycetizoa” phylum. We aimed to classify images into two classes: “Myxomycetes” and “Protosteliomycetes”. However, the available images, 195 for “Myxomycetes” and 2 for “Protosteliomycetes”, posed a significant challenge.

Due to the extreme class imbalance, our models achieved a predictably high accuracy of 100% on the test set and 97.73% on the training set, as they learned to classify all images as “Myxomycetes”. Additional model generation for the “Protozoa” kingdom would not yield significant insights or improvements. This demonstrates the challenges of working with imbalanced datasets and the need for balanced data for reliable model development.

3.3.5 Hierarchy AI models

Bioimage classification presents multiple complex challenges. Calling for innovative solutions, we introduce a novel approach: a hierarchical AI system. This system integrates multiple AI models, each with unique contributions to classifying bioimages across various taxonomic ranks.

Earlier sections outlined the creation of AI models tailored for different taxonomic ranks, from kingdom to phylum and class. We envision a network of interconnected models, forming a comprehensive hierarchy. This is not just an intellectual pursuit but a practical, scalable blueprint.

Our methodology guides a bioimage through taxonomic ranks utilizing the most suitable models at each level. This approach offers a comprehensive classification pathway and enhances overall accuracy by harnessing each model's unique strengths.

In the following sections, we delve into this hierarchical AI system. We detail the process of combining these models into a holistic hierarchy, explain how overall accuracies are computed for each bioimage based on its system journey, and evaluate our models' performance within this hierarchical structure using new image sets.

Hierarchy generation

Biology's taxonomic hierarchy inspired our structure of models as we mirrored its inherent organization. Using our HierarchyAI class, we arranged various standalone models into a coherent hierarchical setup. This class computes accuracy for each potential classification route, generating predictions, classifications suggested by each model along the pathway, and an overall confidence percentage.

Our hierarchical structure employs an adjustable JSON that houses the full model hierarchy, each model reflecting a different taxonomic rank. The structure's flexibility is remarkable: hierarchy adjustments only require JSON modifications, bypassing code changes. The model output at one level directs the next level's model, navigating the image through the classification tree.

We have illustrated the model hierarchy as a network graph (Figure 9) for comprehension. Each node is a model, with directed edges symbolizing the transition from one model to another. Node colors represent taxonomic ranks, and model accuracy is displayed below the model's name.

This model successively refines the classification level until it hits the target rank, like the phylum level for “Animalia”, “Fungi”, and “Plantae”. Each model's accuracy is logged in the structure, assisting in calculating the cumulative accuracy of the hierarchy pathway.

A prediction output consists of three components:

1. A list of dictionaries outlining the predicted taxon rank and value, and the confidence level for each model in the pathway.
2. Cumulative confidence for the final prediction, computed as the product of confidences across the pathway.
3. Overall accuracy, calculated as the product of the accuracies of models used in the pathway.

These components offer a thorough understanding of the predicted results. The list of dictionaries illuminates the performance of each model, while total confidence and accuracy quantifies the hierarchical model's overall reliability.

Quantifying individual and total accuracy helps evaluate our hierarchical model. Individual accuracies highlight performance at each level, indicating improvement areas and efficient classification points. Total accuracy shows the model's overall predictive capability, reflecting the cumulative performance of all models within a specific classification pathway.

Computing the confidence level is crucial. While accuracy quantifies the model's correctness, confidence measures certainty in each prediction. Confidence for each model is determined by the softmax probabilities from the model's final layer, numerically expressing certainty about its classifications (Wu, Y. et al., 2022).

Integrating accuracy and confidence provides a balanced perspective, making our image classification approach robust and insightful.

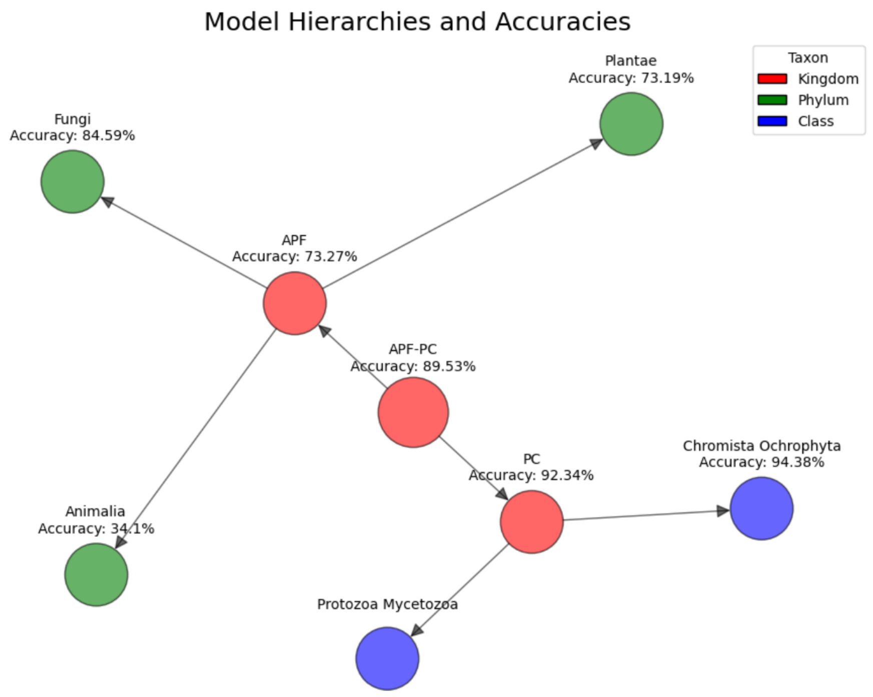


Figure 9. Network graph illustrating the hierarchical configuration of our classification models.

Hierarchy testing

We began with a HierarchyAI class instance, loaded with a JSON embodying our model hierarchy. Next, we assembled a varied image dataset, two images for each possible HierarchyAI prediction result, totaling 50 images from the internet.

Each image's actual labels were identified through a preprocessing step, parsing the image's file name based on the dash character. Resulting labels represented kingdom, phylum, and class associated with the image. Phylum and class were marked as "None" when unavailable, whereas the kingdom was always present.

Predictions were generated using the HierarchyAI instance and preprocessed images, resulting in a series of outcomes for each image. Predicted kingdom, phylum, and class were extracted and integrated into a dataframe. This dataframe also held their actual counterparts and the associated confidence and accuracy metrics for each prediction.

The dataframe was organized as follows: actual kingdom, actual phylum, actual class, predicted kingdom, predicted phylum, predicted class, confidence, and accuracy.

To evaluate HierarchyAI's performance, we crafted a graphical representation (Figure 10), showcasing classification performance across taxonomic ranks. The kingdom was accurately identified in 30 of the 50 images. This achievement can be credited to the three-model system at the kingdom level, distinguishing features across kingdoms. However, accuracy dropped at the phylum level, correctly classifying 12 of the 50 images. This could be due to the single-model system at this level for each kingdom, limiting feature recognition across phyla within a kingdom.

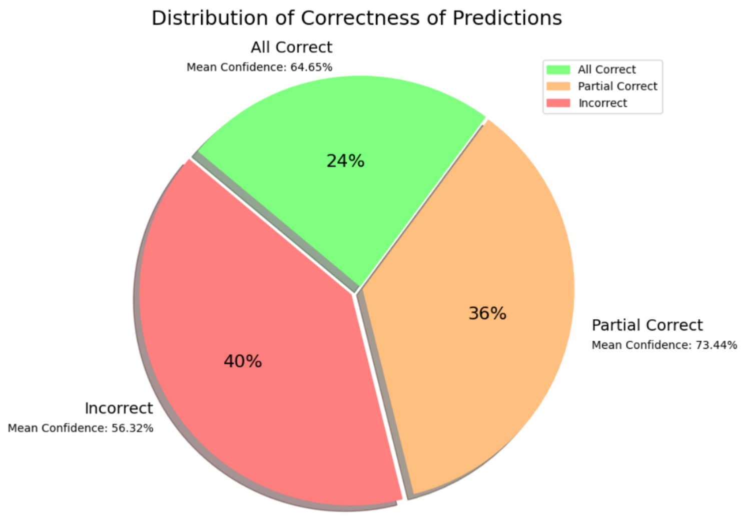


Figure 10: Chart of the distribution of prediction accuracy of the hierarchy AI model across various taxonomic ranks.

Despite these limitations, the test phase validated HierarchyAI's potential. Our goal is demonstrating the methodology rather than achieving maximum prediction accuracy. HierarchyAI's ability to integrate insights from individual AI models into a system offering extensive taxonomic predictions at various levels is commendable. It offers accurate kingdom rank predictions and detailed taxon prediction, confidence, and accuracy data, laying a robust base for future taxonomy classification advancements.

3.4 Models API

While there are existing AI model management services, we decided to develop a novel API, designed to provide functionalities and benefits that go beyond those found in standard service offerings. In comparison to other approaches, our Models API expands the scope of interaction with AI models in significant ways:

1. Custom model registration and management: Our API encourages a dynamic, user-driven ecosystem of AI models. Unlike using a fixed model catalog, we offer the ability to register, modify, delete, and retrieve AI models provided by third parties without modification. This flexibility promotes innovation and personalizes the user experience to an unprecedented degree.
2. Enhanced image classification: We have enhanced the capability of model application to images, extending beyond mere inference and prediction. This advancement elevates the platform's classification capabilities, offering a broader range of use-cases for users.
3. Performance metrics management: We provide the functionality to set, update, and delete performance metrics. This feature contributes to informed model selection and gives users greater control over their tasks, fostering a more user-informed environment.
4. Interactive user feedback management: Our API promotes an interactive, community-driven platform, enabling users to generate, retrieve, and modify feedback for specific images and models, aiding in the continuous refinement of models.
5. Reputation management: Our unique reputation system is designed to encourage high-quality contributions and provide a quantitative measure of user input's impact in the creation of hybrid AI models.

The design of our Models API is based on adaptability and scalability. With direct integration with the NoSQL database, it offers reliable and persistent data handling. Our approach aims to provide a seamless experience of model management, handling a multitude of AI models, and accommodating a wide range of user scenarios.

This Models API does not merely manage models but creates a dynamic ecosystem that promotes innovation, flexibility, and a community-driven approach. By focusing on these facets, our Models API offers a fresh and highly adaptable approach to AI model management. It provides a powerful and complementary approach with respect to other initiatives, e.g., DEEP as a Service (DEEP as a Service 2023).

3.4.1 API Endpoints

The API endpoints offer several enhancements and can effectively be utilized to integrate AI models into platforms such as MINKA. Specifically, they provide the following benefits:

1. Model registration and management: Users can register, alter, delete, and access AI models, cultivating a diverse model ecosystem.
2. Model application to images: Users can apply a chosen model to images in any standard format, significantly improving classification capabilities.
3. Model performance metrics management: Users can set, modify, and remove performance metrics, aiding informed model selection for various tasks.
4. User feedback management: Users can create, access, and modify feedback for specific images and models, creating a useful feedback loop to fine-tune model performance.
5. Reputation management: A reputation system, based on users' successful classification rates, encourages quality contributions and determines the influence of users' inputs in creating hybrid AI models.

The NoSQL database connects directly with the API, ensuring dependable data management. The resulting endpoints, which are both robust and efficient, facilitate the process of image upload and processing using one of the integrated AI models.

In sections below, we introduce an endpoint for hierarchy AI models, with its definition tied to the hierarchy specific form and integrated models, rather than being dynamically defined.

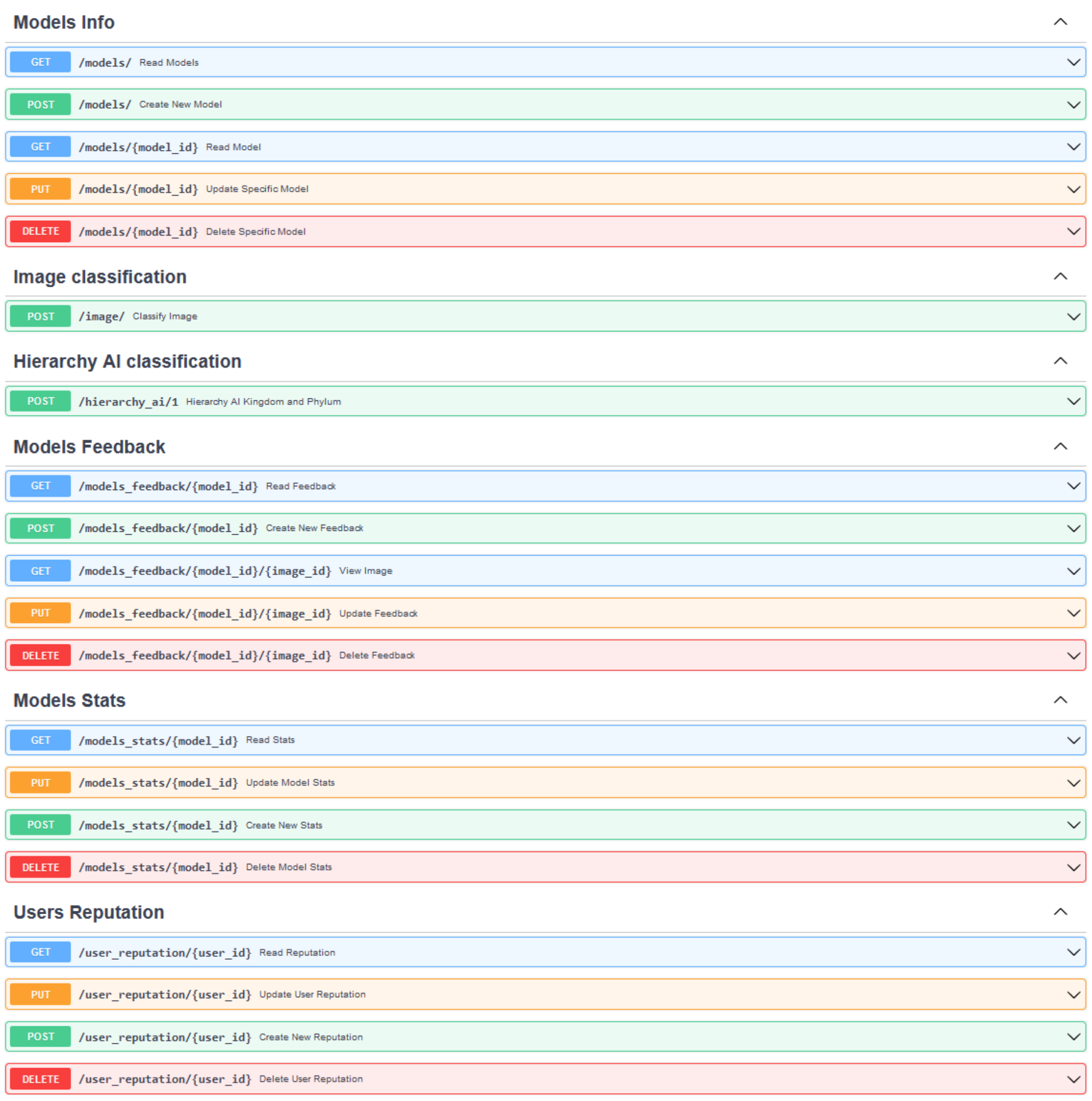
The system employs strict consistency measures designed for various scenarios, including changes in connectivity and database schema. These measures prevent disruptions and data loss from database collections generated by GET, PUT, POST, and DELETE requests. This comprehensive approach ensures that the API endpoints are ready for integration into participatory platforms such as MINKA.

Figure 11 highlights the key functions of the API. This interface shows all the available endpoints and the methods associated with each of them, offering a comprehensive view of the API interaction capabilities.

Model registration and information

The AI model registration and management feature allows users to extend the platform's functionalities and customize their image classification tasks, driving innovation and broadening the AI model ecosystem.

The model registration feature equips the platform with a structured approach to handle various models. The Model and PostModel classes define the AI model structure encapsulating the core properties: title, description, classification level, taxon, supported image formats, accuracy, and API details (URL and key). The automatically generated "id" attribute by the NoSQL database serves as a unique model identifier, essential for referencing models in other endpoints.



Models Info

GET /models/ Read Models

POST /models/ Create New Model

GET /models/{model_id} Read Model

PUT /models/{model_id} Update Specific Model

DELETE /models/{model_id} Delete Specific Model

Image classification

POST /image/ Classify Image

Hierarchy AI classification

POST /hierarchy_ai/1 Hierarchy AI Kingdom and Phylum

Models Feedback

GET /models_feedback/{model_id} Read Feedback

POST /models_feedback/{model_id} Create New Feedback

GET /models_feedback/{model_id}/{image_id} View Image

PUT /models_feedback/{model_id}/{image_id} Update Feedback

DELETE /models_feedback/{model_id}/{image_id} Delete Feedback

Models Stats

GET /models_stats/{model_id} Read Stats

PUT /models_stats/{model_id} Update Model Stats

POST /models_stats/{model_id} Create New Stats

DELETE /models_stats/{model_id} Delete Model Stats

Users Reputation

GET /user_reputation/{user_id} Read Reputation

PUT /user_reputation/{user_id} Update User Reputation

POST /user_reputation/{user_id} Create New Reputation

DELETE /user_reputation/{user_id} Delete User Reputation

Figure 11. Graphical interface of the AI models API.

The introduced operations include:

- Create: Users can register new models, with the system safeguarding against duplicates based on "api url" and "title". Upon successful registration, the system returns the model's details.

- Delete: Users can delete a model using the unique “ id”. If the model is not found, an error message is returned.
- Read: By providing the model’s “ id”, users can access model details, with error handling for cases where the model is not found.
- Update: Users can modify registered model details, enabling the platform to adapt to changes like improvements in model performance or changes in supported image formats.

The structured model design and comprehensive operations empower the platform that integrates them with enhanced capabilities and robust interaction with AI models.

Image classification

Our Image Classification endpoint currently serves as a foundational tool. It classifies images using specific AI models registered on the platform. Here is a brief overview of the endpoint’s operation:

1. Model Verification: Verifies the existence of the user-specified “model id” in the database.
2. Image Processing: The user-uploaded image is resized to a standard dimension of 224x224 pixels.
3. Mock Classification: The endpoint returns a classification result detailing the output of the selected AI models.

By returning the processed image and classification results, the endpoint has seamless interaction between users, images, and AI models that may be integrated into MINKA.

Models stats

The Models Stats component serves to manage statistical data about AI model performance. It is a protected component with secure access since these statistics are crucial for informed decision- making during image classification and are assigned internally.

The component registers and manages three core metrics for each model:

- Accuracy: Represents how often the model is correct in its predictions. High accuracy indicates the model generally provides reliable results.
- Precision: Measures how many of the model’s positive predictions were correct. High precision suggests that when the model predicts a specific class, it is likely accurate.

- Recall: Assesses how many actual positive instances the model correctly identified. High recall suggests the model is good at detecting positive instances.

The Models Stats component's operations are as follows:

- Read: Retrieves stats for a specified model. This function is the only one accessible to the public.
- Create: Stores statistical data for a specified model.
- Delete: Removes stats for a specified model.
- Update: Modifies stored stats for a specified model.

By maintaining these metrics, Models Stats ensures quality and reliability.

Model feedback

The Model Feedback component is crucial for a hybrid AI system, which combines AI and human intelligence. It collects users feedback on AI model predictions, promoting a cooperative learning atmosphere where AI models can adapt based on human expertise.

The component collects feedback data like the correct image category, the model's predicted category, and the image itself. This data is stored and added to existing feedback or used to create a new feedback entry.

Here are the operations managed by the Model Feedback component:

- Create: Takes the model ID, the correct and predicted values, and the image to create or append a new feedback entry in the database.
- Read: Retrieves feedback data for a specific model returning the entire feedback entry.
- Update: Accepts an updated feedback item and the ID of the associated image to update
- the specific feedback entry in the database.
- Delete: Removes feedback for a specific model and image by deleting the corresponding database entry.

This feedback channel enables continuous model refinement, contributing to model performance improvement and an evolutionary development process driven by user engagement.

Ultimately, feedback data combined with model stats and user reputation scores is vital for decisions regarding model re-training. This integrated approach is at the heart of our hybrid AI system, serving to enhance not only the accuracy of AI models but also the overall reliability of the platform.

In short, the Model Feedback system encourages a unique human-AI collaboration in line with our citizen science approach.

Users reputation

Lastly, we have incorporated the Users Reputation endpoint into our API. This feature highlights how our hybrid AI system uses the reputation of users to assess the feedback they give to the AI model. This fits perfectly with our citizen science approach, where we value contributions from the user community. At the same time, we maintain a strong system to ensure the quality of information provided.

The users reputation feature is crucial for assessing the reliability of users feedback and represents a quality control mechanism for user inputs. Every user’s credibility is defined by their reputation score, a measure based on past engagements and feedback accuracy. This reputation system works hand in hand with the Model Feedback system, offering a way to evaluate user feedback and thereby fine-tuning the AI models. Users with higher reputation scores gain more credibility for their feedback.

The UserReputation model, which tracks a user's reputation, embodies this feature. It considers five principal attributes: reputation, total entries, correct predictions, incorrect predictions, and unconfirmed predictions. These metrics offer a well-rounded view of a user’s performance and standing within the platform.

Here is a rundown of the key operations of the users reputation system:

- Create: Establishes a reputation entry for a user ID with the five required attributes.
- Read reputation: Fetches a user’s reputation using their ID which aids in evaluating the credibility of the user’s feedback in the hybrid AI system.
- Update: Accepts an updated reputation for the users as it evolves over time, reflective of their interactions and feedback.
- Delete: Erases a user’s reputation record.

User reputation plays a pivotal role in our hybrid AI system. When the AI model’s classification differs from the user’s feedback, the user’s reputation serves as a decision point. High reputation could tip the final classification towards the user’s feedback, whereas lower reputation could move the decision towards the AI model’s initial prediction.

To sum up, by considering users reputation, we ensure the effective harnessing of collective intelligence. This leads to a dynamic ecosystem where users actively participate in improving the AI models, resulting in a continually evolving, flexible, and user-focused platform.

3.4.2 Model integration

Model integration is a cornerstone of AMOVALIH, amplifying its learning capacity, adaptability, and ability to tackle complex problems. This process involves the incorporation of both local and external models, thereby boosting the system's capabilities and versatility.

In previous sections, we described the process for generating local models tailored to address specific problems and evaluated for robust performance. Of these, the hierarchy AI model is particularly noteworthy for its superior performance and unique problem-solving approach, offering significant potential enhancements to our system. These local models play a crucial role in testing the system integration, offering proof of concept for the project, as they provide a controlled environment where we can examine and refine the system's ability to integrate, interact with, and retrain models based on user's feedback, which is key to enabling our hybrid AI vision.

Nevertheless, external models are the true engines of value for our API. They allow us to tap into a wealth of knowledge and expertise beyond our local models, capitalizing on different perspectives, techniques, and solutions that our locally developed models may not encapsulate.

After integrating models, we conduct rigorous testing to ensure functionality and correct output. The following sections provide a detailed view of each aspect of model integration.

Local models

Our local models are critical to the system, featuring tailored solutions for complex tasks. We have integrated our four CNN models, each providing robust classification of various biological kingdoms.

- Kingdom ALL CNN: Predicts five different kingdoms with an accuracy of 59.45%.
- Kingdom APF-PC CNN: Classifies kingdoms into two broader groups with an accuracy of 89.53%.
- Kingdom APF CNN: Detailed classification between three kingdoms with an accuracy of 73.27%.

- Kingdom PC CNN: Detailed classification between two kingdoms with an exceptional accuracy of 92.34%.

These local models offer benefits like flexibility, as we control the model's architecture, training, and used data. They are relevant to our use-cases, fine-tuned for the data they will encounter, and can be quickly modified as needed.

Integration involves verifying the local model file, loading the model using its relative path, and predicting the classification from the uploaded image. Any failure triggers an HTTPException detailing the issue, aiding maintenance and debugging.

External models

Apart from local models, we have incorporated external models to broaden our system's capabilities, tapping into a wider pool of AI insights and techniques. A case in point is our use of the Pl@ntNet API model, a plant classification model that, while not stating its accuracy, covers an expansive range of classifications. This integration is a proof-of-concept to demonstrate the viability of the approach to incorporate third-party models, even outside of the scope of operational marine biology.

External models allow us to capitalize on cutting-edge AI research and methods from diverse sources. They also help extend our classification capabilities beyond our model training, and are often extensively validated and tested, ensuring reliable performance.

The integration process is similar to that of local models, but with more complex image loading and header generation for correct image transmission to the external API. Any failure in fetching the model or making the POST request triggers an "HTTPException", aiding debugging and maintenance.

Hierarchy AI models

Implementing hierarchy AI models marks a major enhancement to image classification capabilities. These models employ a hierarchical approach, passing the output of one model as input to the next. This reflects the hierarchical nature of taxonomic classification (Kingdom, Phylum, Class, etc.), although automating or making it dynamic is challenging due to each model's unique characteristics.

Despite this complexity, Hierarchy AI models greatly improve predictions accuracy and provide more detailed classification results, enabling the integration of diverse AI models. To accommodate this, we have created a dedicated endpoint, POST /hierarchy ai/1, for the hierarchy AI classification of kingdom and phyla.

While implementing these models is complex, they represent a significant advancement. They facilitate accurate classifications while capturing biological taxonomy's hierarchical nature. As we add more models, this system can be expanded and adjusted, helping deliver even more precise classification results.

3.4.3 Testing the integrated proof-of-concept models

Testing the integrated models is crucial to ensure they work correctly and effectively within the system. As a reminder, there is an online version that can be used for testing these models and other functionalities at the Models API GitHub.

Firstly, we inspect the /models endpoint through a GET request. This returns a list of all models, including both local and external ones. For instance, the "Kingdom All CNN" model is a local model, while the "Pl@ntNet API" model is an external one:

```
Kingdom All:
{
    ...
    "title": "Kingdom ALL CNN",
    "description": "This model returns a predicted value for the kingdoms: 0 - Animalia, 1 - Chromista, 2 - Fungi, 3 - Plantae, 4 - Protozoa.",
    "classification_level": "Kingdom",
    ...
    "accuracy": 59.45,
    "api_url": "cnn_kingdom_model.h5",
    "api_key": "local"
}
```

```
Pl@ntNet:
{
    ...
    "title": "Pl@ntNet API",
    "description": "API for ALL plants classification",
    "classification_level": "all",
    ...
    "api_url": "https://my-api.plantnet.org/v2/identify/all?include-related-images=false&no-reject=false&lang=en&api-key=",
    "api_key": "XXX"
}
```

These responses contain essential details about each model like their origin, the taxonomic rank they classify, and their accuracy.

Next, we test the /image endpoint designed to accept an image input and return the classification result. Below are example outputs for both local and external models:

```
Medusa image sent to the Kingdom All model:
{
  "prediction": 0,
  "details": "This model returns a predicted value for the kingdoms
   : 0 - Animalia, 1 - Chromista, 2 - Fungi, 3 - Plantae, 4 -
   Protozoa."
}
```

```
Bangiaceae (Rhodophyta phylum) image sent to the Pl@ntNet model:
{
    ...
    "bestMatch": "Posidonia oceanica (L.) Delile",
    "results": [
      {
        "score": 0.15351,
        "species": {
          "scientificNameWithoutAuthor": "Posidonia oceanica",
          ...
          "commonNames": [
            "Mediterranean tapeweed",
            ...
          ],
          ...
        },
        ...
      },
      ...
    ]
}
```

Lastly, we test the /hierarchy ai/1 endpoint. It should return a list of classification results for each taxonomic rank:

```
Basidiomycota image (Fungi) sent to the hierarchy AI model:
[
  {
    "taxon": "Kingdom",
    "result": "Fungi",
    "confidence": 100
  },
  {
    "taxon": "Phylum",
    "result": "Basidiomycota",
    "confidence": 100
  }
]
```

3.5 Hybrid AI

Hybrid AI is an innovative approach that harmoniously combines the computational power of AI with the knowledge and expertise of human users. The result is a robust, adaptive system that continually enhances its predictive performance based on users feedback.

Our hybrid AI system revolves around a mechanism that leverages users feedback along with their reputation and model performance statistics. This allows our system to adaptively learn and retrain the models, thereby evolving the predictive capabilities of the model over time. Initially, we only use user feedback for retraining the model, but this paves the way for more complex interactions that include reputation and model stats.

In the following sections, we will discuss how we utilize user feedback for adaptive learning, followed by an overview of the model retraining process. Our ultimate goal with this hybrid AI system is to seamlessly blend artificial and human intelligence, resulting in a tool that not only aids in identification and classification tasks but also continuously learns and improves with each interaction, maintaining relevance and accuracy.

3.5.1 Leverage users' feedback, reputation, and model stats

Our hybrid AI system's key strength is its ability to learn and adapt continuously, facilitated by harnessing users feedback, reputation scores, and model performance statistics. While we value all feedback, the system primarily focuses on instances where the model's prediction conflicts with the user's feedback, i.e., incorrect predictions, as these instances offer great potential for learning and improving the model.

The users reputation scores, ranging from 1 (poor) to 10 (excellent), reflect their historical accuracy in providing feedback, with feedback from users having a higher reputation score considered more reliable. Model stats such as accuracy, precision, and recall provide a quantitative measure of the model's current performance.

We need to calculate a decision value (V) based on these metrics, helping us decide whether to incorporate the user's feedback for model retraining. This decision value takes into account both the user's reputation and the model's stats, and is calculated as follows:

$V = (R \cdot W_1) - ((A \cdot W_2) + (P \cdot W_3) + (C \cdot W_4))$

where:

- R represents the user's reputation score, between 1 and 10.
- A represents the model's accuracy, between 0 and 100.
- P represents the model's precision, between 0 and 100.
- C represents the model's recall, between 0 and 100.
- W_i are weights that range from 0 to 1.

The weights W_i can vary from 0 to 1, depending on the importance assigned to each metric. For instance, a higher value can be set for W_1 if user reputation is considered more important. We ensure that the sum of all weights equals 1 through normalization. Consequently, the reputation, accuracy, precision, and recall scores need to be expressed as ratios between 0 and 1 for accurate calculations or further adjustment of this function.

This decision value function is designed to balance user inputs against model performance. For instance, if a high-reputation user's feedback conflicts with a well-performing model, the decision value will be around or greater than 5, suggesting a need for manual review or retraining based on user feedback. Conversely, if a user with a low reputation score contradicts a well-performing model, the decision value will be significantly below 5, prompting us to rely on the model's prediction.

3.5.2 Model retraining and adaptive learning

The integration of adaptive learning and model retraining is a significant innovation within the sphere of AI-enabled citizen science projects. This innovation enhances project capabilities by facilitating continuous improvements in AI models via user feedback and a decision value computation.

The application of this retraining and adaptive learning facilitates dynamic adaptation of registered AI models to changing data. The models learn from previous errors and evolve to augment their real-time predictive capabilities. This process harnesses users feedback to enhance model performance and accuracy, leading to the evolution of a responsive AI system that grows with the influx of new data.

The decision value determines whether the model will be immediately retrained ($V > 5$) or if the image will be stored for an expert to assess the quality of the feedback for retraining ($V <= 5$). Currently, the decision value function consistently returns 7, ensuring all feedback contributes directly to retraining.

For retraining, all image data is stored in a storage system and they are resized and stored for subsequent model retraining, ensuring the complex system's smooth operation and robust response capabilities. Errors encountered during the feedback process or model retraining are swiftly resolved, with detailed error responses facilitating swift and efficient debugging.

In practice, the retraining process stores user feedback, calculates a decision value where if it is above 5, preprocesses the associated image, maps the correct value to its categorical equivalent, and utilizes this data to retrain and save the model.

For a practical example of this retraining process, consider a scenario where a "Fungi" image was misclassified as "Plantae" by the Kingdom AI model. Below are the details of this process:

Initial image classification request:

```
curl -X 'POST'
'http://API_HOSTNAME:8000/image/?model_id=6472204f4d944c97189313c3'
-H 'accept: application/json'
-H 'Content-Type: multipart/form-data'
-F 'image_data=@72951_95657.jpg;type=image/jpeg'
```

Initial image classification response:

```
{
    "prediction": 3,
    "details": "This model returns a predicted value for the
  kingdoms: 0 - Animalia, 1 - Chromista, 2 - Fungi, 3 - Plantae, 4
   - Protozoa."
}
```

Feedback submission for correct classification:

```
curl -X 'POST'
'http://API_HOSTNAME:8000/models_feedback/6472204f4d944c97189313c3'
-H 'accept: application/json'
-H 'Content-Type: multipart/form-data'
-F 'correct_value=2'
-F 'model_value=3'
-F 'image=@72951_95657.jpg;type=image/jpeg'
```

Feedback submission response:

```
{
    "message": "AI model retrained successfully."
}
```

Retesting the image classification after retraining:

```
curl -X 'POST'
'http://API_HOSTNAME:8000/image/?model_id=6472204f4d944c97189313c3'
-H 'accept: application/json'
-H 'Content-Type: multipart/form-data'
-F 'image_data=@72951_95657.jpg;type=image/jpeg'
```

Retesting response:

```
{
    "prediction": 2,
    "details": "This model returns a predicted value for the
  kingdoms: 0 - Animalia, 1 - Chromista, 2 - Fungi, 3 - Plantae, 4
   - Protozoa."
}
```

Now, the model accurately classifies the image under the right category. This example demonstrates how our hybrid AI leverages inaccurate predictions as learning opportunities, thereby progressively improving the model's accuracy and performance over time.

References

Affonso, C. et al. (2017). “Deep learning for biological image classification”. In: Expert Systems with Applications 85, pp. 114–122. issn: 0957-4174. doi: 10.1016/j.eswa.2017.05.039.

Ahmad, K. et al. (2022). “Developing future human-centered smart cities: Critical analysis of smart city security, Data management, and Ethical challenges”. In: Computer Science Review 43, p. 100452. doi: 10.1016/j.cosrev.2021.100452.

AI, HLEG (2019). High-level expert group on artificial intelligence. Ethics guidelines for trustworthy AI. https://www.aepd.es/sites/default/files/2019-09/ai-definition.pdf.

Albumentations (2023). https://albumentations.ai/.

Alom, M. Z. et al. (2018). “The history began from AlexNet: A comprehensive survey on deep learning approaches”. In: arXiv preprint arXiv:1803.01164 [cs.CV]. doi: 10.48550/arXiv.1803.01164.

Amato, G. and F. Falchi (2010). “kNN based image classification relying on local feature similarity”. In: SISAP ’10: Proceedings of the Third International Conference on SImilarity Search and APplications, pp. 101–108. doi: 10.1145/1862344.1862360.

Bailer, C. et al. (2018). “Fast Feature Extraction with CNNs with Pooling Layers”. In: arXiv:1805.03096 [cs.CV]. doi: 10.48550/arXiv.1805.03096.

Barlow, H. B. (1989). “Unsupervised learning”. In: Neural Computation 1.3, pp. 295–311. doi: 10.1162/neco.1989.1.3.295. — (1989). “Unsupervised Learning”. In: Neural Computation 1.3, pp. 295–311. doi: 10.1162/neco.1989.1.3.295.

Bartkowski, B., N. Lienhoop, and B. Hensju ̈rgens (2015). “Capturing the complexity of biodiversity: A critical review of economic valuation studies of biological diversity”. In: Ecological Economics 113, pp. 1–14. doi: 10.1016/j.ecolecon.2015.02.023. url: https://doi.org/10.1016/j.ecolecon.2015.02.023.

Basha, S. H. S. et al. (2020). “Impact of fully connected layers on performance of convolutional neural networks for image classification”. In: Neurocomputing 378, pp. 112–119. doi: 10.1016/j.neucom.2019.10.008.

Bernal, J., J. Sa ́nchez, and F. Vilarin ̃o (2013). “Impact of image preprocessing methods on polyp localization in colonoscopy frames”. In: Annu Int Conf IEEE Eng Med Biol Soc, pp. 7350–7354. doi: 10.1109/EMBC.2013.6611256.

Bharati, S. et al. (2022). “Deep Learning for Medical Image Registration: A Comprehensive Review”. In: International Journal of Computer Information Systems and Industrial Management Applications 14, pp. 173–190. issn: 2150-7988.

Bilbao, I. and J. Bilbao (2017). “Overfitting problem and the over-training in the era of data: Particularly for Artificial Neural Networks”. In: 2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS), pp. 173–177. doi: 10.1109/INTELCIS.2017.8260032.

Bloice, M. D., C. Stocker, and A. Holzinger (2017). Augmentor: An Image Augmentation Library for Machine Learning. arXiv: 1708.04680 [cs.CV]. url: https://arxiv.org/abs/1708. 04680.

Bora, D. J., A. K. Gupta, and F. A. Khan (2015). “Comparing the Performance of LAB* and HSV Color Spaces with Respect to Color Image Segmentation”. In: International Journal of Emerging Technology and Advanced Engineering 5.2. doi: 10.48550/arXiv.1506.01472.

Breiman, L. (2001). “Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author)”. In: Statistical Science 16.3, pp. 199–231. doi: 10.1214/ss/1009213726.

Bryan, N. J. (2020). “Impulse Response Data Augmentation and Deep Neural Networks for Blind Room Acoustic Parameter Estimation”. In: ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. doi: 10.1109/ICASSP40776.2020.9052970.

Cai, Jie et al. (2018). “Feature selection in machine learning: A new perspective”. In: Neurocomputing 300, pp. 70–79. doi: 10.1016/j.neucom.2017.11.077.

Carter, J. V. et al. (2016). “ROC-ing along: Evaluation and interpretation of receiver operating characteristic curves”. In: Surgery 159.6, pp. 1638–1645. doi: 10.1016/j.surg.2015.12.029.

Caruana, R. and A. Niculescu-Mizil (2006). “An empirical comparison of supervised learning algorithms”. In: ICML ’06: Proceedings of the 23rd international conference on Machine learning, pp. 161–168. doi: 10.1145/1143844.1143865.

Chassagnon, G., M. Vakalopolou, and N. Paragios (2020). “Deep learning: definition and perspectives for thoracic imaging”. In: Eur Radiol 30, pp. 2021–2030. doi: 10.1007/s00330-019-06564-3.

Chou, J.-S., M.-Y. Cheng, and Y.-W. Wu (2013). “Improving classification accuracy of project dispute resolution using hybrid artificial intelligence and support vector machine models”. In: Expert Systems with Applications 40.6, pp. 2263–2274. issn: 0957-4174. doi: 10.1016/j.eswa. 2012.10.036.

Christlein, V. et al. (2019). “Deep Generalized Max Pooling”. In: 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 1090–1096. doi: 10.1109/ICDAR.2019. 00177.

Corchado, J. M. and J. Aiken (2002). “Hybrid artificial intelligence methods in oceanographic fore- cast models”. In: IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 32.4, pp. 307–313. doi: 10.1109/TSMCC.2002.806072.

Cui, S. et al. (2021). “A stacking-based ensemble learning method for earthquake casualty prediction”. In: Applied Soft Computing 101, p. 107038. doi: 10.1016/j.asoc.2020.107038.

Dafonte, C. et al. (2020). “A Blended Artificial Intelligence Approach for Spectral Classification of Stars in Massive Astronomical Surveys”. In: Entropy 22.5, p. 518. doi: 10.3390/e22050518.

Dayan, P. and Y. Niv (2008). “Reinforcement learning: The good, the bad and the ugly”. In: Current Opinion in Neurobiology 18.2, pp. 185–196. doi: 10.1016/j.conb.2008.08.003.

DEEP as a Service (2023). https://deepaas.deep-hybrid-datacloud.eu.

Desai, P., J. Pujari, Akhila, et al. (2021). “Impact of Multi-Feature Extraction on Image Retrieval and classification Using Machine Learning Technique”. In: SN Computer Science 2.153. doi:10.1007/s42979-021-00532-9.

Ding, X. (2017). “Texture Feature Extraction Research Based on GLCM-CLBP Algorithm”. In: doi: 10.2991/emim-17.2017.36.

Elharrouss, O. et al. (2022). “Backbones-Review: Feature Extraction Networks for Deep Learning and Deep Reinforcement Learning Approaches”. In: arXiv preprint arXiv:2206.08016 [cs.CV]. doi: 10.48550/arXiv.2206.08016.

Engelen, J.E. van and H.H. Hoos (2020). “A survey on semi-supervised learning”. In: Machine Learning 109, pp. 373–440. doi: 10.1007/s10994-019-05855-6.

Fawzi, A. et al. (2016). “Adaptive data augmentation for image classification”. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3688–3692. doi: 10.1109/ICIP. 2016.7533048.

Github (2023). Augmentor. https://github.com/mdbloice/Augmentor.

Guo, L. et al. (2022). “A deep learning-based hybrid artificial intelligence model for the detection and severity assessment of vitiligo lesions”. In: Annals of Translational Medicine 10.10, p. 590. doi: 10.21037/atm-22-1738.

Hall, E. L. et al. (1971). “A Survey of Preprocessing and Feature Extraction Techniques for Radiographic Images”. In: IEEE Transactions on Computers C-20.9, pp. 1032–1044. doi: 10.1109/T-C.1971.223399.

Hickman, E. and M. Petrin (2021). “Trustworthy AI and Corporate Governance: The EU’s Ethics Guidelines for Trustworthy Artificial Intelligence from a Company Law Perspective”. In: Eur Bus Org Law Rev 22, pp. 593–625. doi: 10.1007/s40804-021-00224-0.

Hira, Z. M. and D. F. Gillies (2015). “A Review of Feature Selection and Feature Extraction Methods Applied on Microarray Data”. In: Advances in Bioinformatics 2015. doi: 10.1155/2015/198363.

Huang, J., S. R. Kumar, and R. Zabih (1998). “An Automatic Hierarchical Image Classification Scheme”. In: Proceedings of the Sixth ACM International Conference on Multimedia, MULTIMEDIA ’98, pp. 219–228. doi: 10.1145/290747.290774.

Ibrahim, K. S. M. H. et al. (2022). “A review of the hybrid artificial intelligence and optimization modelling of hydrological streamflow forecasting”. In: Alexandria Engineering Journal 61.1, pp. 279–303. doi: 10.1016/j.aej.2021.04.100.

Kaelbling, L. P., M. L. Littman, and A. W. Moore (1996). “Reinforcement learning: A survey”. In: Journal of Artificial Intelligence Research 4, pp. 237–285. doi: 10.1613/jair.301.

Kazim, E. and A. S. Koshiyama (2021). “A high-level overview of AI ethics”. In: Patterns 2.9, p. 100314. doi: 10.1016/j.patter.2021.100314.

Kebonye, N. M. (2021). “Exploring the novel support points-based split method on a soil dataset”. In: Measurement 186, p. 110131. doi: 10.1016/j.measurement.2021.110131.

Krstinic, D. et al. (2020). “Multi-label classifier performance evaluation with confusion matrix”. In: SAIM, SIPM, ACSIT, FCST, ICITE - 2020. CS IT - CSCP 2020, pp. 01–14. doi: 10.5121/csit.2020.100801.

Kumar, G. and P.K. Bhatia (2014). “A Detailed Review of Feature Extraction in Image Processing Systems”. In: 2014 Fourth International Conference on Advanced Computing & Communication Technologies, pp. 5–12. doi: 10.1109/ACCT.2014.74.

Kumar, R.L., J. Kakarla, B.V. Isunuri, et al. (2021). “Multi-class brain tumor classification using residual network and global average pooling”. In: Multimed Tools Appl 80, pp. 13429–13438. doi: 10.1007/s11042-020-10335-4.

LeCun, Y., Y. Bengio, and G. Hinton (2015). “Deep learning”. In: Nature 521, pp. 436–444. doi: 10.1038/nature14539.

Lepri, B., N. Oliver, and A. Pentland (2021). “Ethical machines: The human-centric use of artificial intelligence”. In: iScience 24.3, p. 102249. doi: 10.1016/j.isci.2021.102249.

Li, J. M., J. M. Bioucas-Dias, and A. Plaza (2013). “Semisupervised Hyperspectral Image Classification Using Soft Sparse Multinomial Logistic Regression”. In: IEEE Geoscience and Remote Sensing Letters 10.2, pp. 318–322. doi: 10.1109/LGRS.2012.2205216.

Liang, Y. et al. (2023). TaskMatrix.AI: Completing Tasks by Connecting Foundation Models with Millions of APIs. arXiv: 2303.16434 [cs.AI]. url: https://arxiv.org/abs/2303.16434.

Liberty, E., K., Lang, and K. Shmakov (2016). “Stratified Sampling Meets Machine Learning”. In: Proceedings of The 33rd International Conference on Machine Learning, pp. 2320–2329. url: https://proceedings.mlr.press/v48/liberty16.html.

Lin, Y. et al. (2011). “Large-scale image classification: Fast feature extraction and SVM training”.

Lins, S., K.D. Pandl, H. Teigeler, et al. (2021). “Artificial Intelligence as a Service”. In: Bus Inf Syst Eng 63, pp. 441–456. doi: 10.1007/s12599-021-00708-w. url: https://doi.org/10.1007/s12599-021-00708-w.

López García, A. (2019). “DEEPaaS API: a REST API for Machine Learning and Deep Learning models”. In: Journal of Open Source Software 4.41, p. 1517. doi: 10.21105/joss.01517.

Loussaief, S. and A. Abdelkrim (2016). “Machine learning framework for image classification”. In: 2016 7th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), pp. 58–61. doi: 10.1109/SETIT.2016.7939841.

Manis, K.T. and S. Madhavaram (2023). “AI-Enabled marketing capabilities and the hierarchy of capabilities: Conceptualization, proposition development, and research avenues”. In: Journal of Business Research 157, p. 113485. doi: 10.1016/j.jbusres.2022.113485.

Menard, S. (2002). Applied Logistic Regression Analysis. SAGE Publications, Inc. url: https://doi.org/10.4135/9781412983433.

Moazamnia, M. et al. (2019). “Formulating a strategy to combine artificial intelligence models using Bayesian model averaging to study a distressed aquifer with sparse data availability”. In: Journal of Hydrology 571, pp. 765–781. issn: 0022-1694. doi: 10.1016/j.jhydrol.2019.02. 011.

Moerland, T. M. et al. (2023). “Model-based Reinforcement Learning: A Survey”. In: Foundations and Trends® in Machine Learning 16.1, pp. 1–118. doi: 10.1561/2200000086.

Moradmand, H., S. M. R. Aghamiri, and R. Ghaderi (2019). “Impact of image preprocessing methods on reproducibility of radiomic features in multimodal magnetic resonance imaging in glioblastoma”. In: Medical Physics 46.3, pp. 1243–1253. doi: 10.1002/acm2.12795.

Murtaza, G., L. Shuib, A.W. Abdul Wahab, et al. (2020). “Deep learning-based breast cancer classification through medical imaging modalities: state of the art and research challenges”. In: Artif Intell Rev 53, pp. 1655–1720. doi: 10.1007/s10462-019-09716-5.

Nanni, L. et al. (2019). “Bioimage Classification with Handcrafted and Learned Features”. In: IEEE/ACM Transactions on Computational Biology and Bioinformatics 16.3, pp. 874–885. doi: 10.1109/TCBB.2018.2821127.

Nasr-Esfahani, E. et al. (2019). “Dense pooling layers in fully convolutional network for skin lesion segmentation”. In: Computerized Medical Imaging and Graphics 78. issn: 0895-6111. doi: 10.1016/j.compmedimag.2019.101658.

Nasteski, V. (2017). “An overview of the supervised machine learning methods”. In: Horizons 4, pp. 51–62. doi: 10.20544/HORIZONS.B.04.1.17.P05.

Ning, X. et al. (2022). “A new generation of ResNet model based on artificial intelligence and few data driven and its construction in image recognition model”. In: Computational Intelligence and Neuroscience 2022, p. 5976155. doi: 10.1155/2022/5976155.

Paymode, A. S. and V. B. Malode (2022). “Transfer learning for multi-crop leaf disease image classification using convolutional neural network VGG”. In: Artificial Intelligence in Agriculture 6, pp. 23–33. doi: 10.1016/j.aiia.2021.12.002.

Perez, L. and J. Wang (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. arXiv: 1712.04621 [cs.LG]. url: https://arxiv.org/abs/1712. 04621.

Pham, H. V. et al. (2020). “Problems and opportunities in training deep learning software systems: an analysis of variance”. In: Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering (ASE ’20), pp. 771–783. doi: 10.1145/3324884.3416545.

Popescu, M. C. and L. M. Sasu (2014). “Feature extraction, feature selection and machine learning for image classification: A case study”. In: 2014 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM), pp. 968–973. doi: 10.1109/OPTIM.2014. 6850925.

Rapport, D.J., R. Costanza, and A.J. McMichael (1998). “Assessing ecosystem health”. In: Trends in Ecology Evolution 13.10, pp. 397–402. issn: 0169-5347. doi: 10 . 1016 / S0169 - 5347(98) 01449-9.

Ray, S. (2019). “A Quick Review of Machine Learning Algorithms”. In: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), pp. 35–39. doi: 10.1109/COMITCon.2019.8862451.

Sagi, O. and L. Rokach (2018). “Ensemble learning: A survey”. In: Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8.4. doi: 10.1002/widm.1249.

Saha, S. et al. (2021). “Hierarchical Deep Learning Neural Network (HiDeNN): An artificial intelligence (AI) framework for computational science and engineering”. In: Computer Methods in Applied Mechanics and Engineering 373, p. 113452. issn: 0045-7825. doi: 10.1016/j.cma. 2020.113452.

Sala, O. E. et al. (2000). “Global Biodiversity Scenarios for the Year 2100”. In: Science 287.5459, pp. 1770–1774. doi: 10.1126/science.287.5459.1770. url: https://www.science.org/ doi/10.1126/science.287.5459.1770.

SDGs (2023). https://sdgs.un.org/goals.

Seventekidis, P. and D. Giagopoulos (2021). “A combined finite element and hierarchical Deep learning approach for structural health monitoring: Test on a pin-joint composite truss structure”.

In: Mechanical Systems and Signal Processing 157, p. 107735. issn: 0888-3270. doi: 10.1016/j.ymssp.2021.107735.

Shekar, G., S. Revathy, and E. K. Goud (2020). “Malaria Detection using Deep Learning”. In: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), pp. 746–750. doi: 10.1109/ICOEI48184.2020.9143023.

Smuha, Nathalie A. (2019). “The EU Approach to Ethics Guidelines for Trustworthy Artificial Intelligence”. In: Computer Law Review International 20.4, pp. 97–106. doi: 10.9785/cri-2019-200402.

Solari, F. et al. (2021). “Accurate Multilevel Classification for Wildlife Images”. In: Computational Intelligence and Neuroscience 2021, p. 6690590. issn: 1687-5265. doi: 10.1155/2021/6690590.

Song, J. et al. (2019). “Multi-layer boosting sparse convolutional model for generalized nuclear segmentation from histopathology images”. In: Knowledge-Based Systems 176, pp. 40–53. issn: 0950-7051. doi: 10.1016/j.knosys.2019.03.031.

Stefenon, S.F., KC. Yow, A. Nied, et al. (2022). “Classification of distribution power grid structures using inception v3 deep neural network”. In: Electrical Engineering 104, pp. 4557–4569. doi: 10.1007/s00202-022-01641-1.

Sun, J. et al. (2020). “New Interpretations of Normalization Methods in Deep Learning”. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34. 4. doi: 10.1609/aaai.v34i04.6046.

Talathi, S. S. and A. Vartak (2016). “Improving performance of recurrent neural network with ReLU nonlinearity”. In: arXiv preprint arXiv:1511.03771v3 [cs.NE]. url: https://arxiv.org/abs/1511.03771.

Tan, Mingxing and Quoc V. Le (2019). “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: CoRR abs/1905.11946. arXiv: 1905.11946. url: http://arxiv.org/abs/1905.11946.

Thomaz, C. E. and G. A. Giraldi (2010). “A new ranking method for principal components analysis and its application to face image analysis”. In: Image and Vision Computing 28.6, pp. 902–913. doi: 10.1016/j.imavis.2009.11.005.

Tilman, D. (2000). “Causes, consequences and ethics of biodiversity”. In: Nature 405, pp. 208–211. doi: 10.1038/35012217. url: https://doi.org/10.1038/35012217.

Trier, Ø. D., A. K. Jain, and T. Taxt (1996). “Feature extraction methods for character recognition - A survey”. In: Pattern Recognition 29.4, pp. 641–662. issn: 0031-3203. doi: 10.1016/0031-3203(95)00118-2.

Vijayan, V. and E. Sherly (2019). “A deep neural architecture for driver drowsiness detection using facial features”. In: Journal of Intelligent Fuzzy Systems 36.3, pp. 1977–1985. doi:10.3233/JIFS-169909.

Wang, P., E. Fan, and P. Wang (2021). “Comparative analysis of image classification algorithms based on traditional machine learning and deep learning”. In: Pattern Recognition Letters 141, pp. 61–67. issn: 0167-8655. doi: 10.1016/j.patrec.2020.07.042.

Wang, Z. and R. S. Srinivasan (2017). “A review of artificial intelligence-based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models”. In: Renewable and Sustainable Energy Reviews 75, pp. 796–808. issn: 1364-0321. doi: 10.1016/j.rser.2016.10.079.

Wu, D. et al. (2021). “Edge-Cloud Collaboration Enabled Video Service Enhancement: A Hybrid Human-Artificial Intelligence Scheme”. In: IEEE Transactions on Multimedia 23, pp. 2208–2221. doi: 10.1109/TMM.2021.3066050.

Wu, Y. et al. (2022). “A deep learning fusion model with evidence-based confidence level analysis for differentiation of malignant and benign breast tumors using dynamic contrast enhanced MRI”. In: Biomedical Signal Processing and Control 72.Part B, p. 103319. doi: 10.1016/j. bspc.2021.103319.

Xiao, Y. et al. (2018). “A deep learning-based multi-model ensemble method for cancer prediction”. In: Computer Methods and Programs in Biomedicine 153, pp. 1–9. doi: 10.1016/j.cmpb.2017. 09.005.

Xu, M. et al. (2023). “A Comprehensive Survey of Image Augmentation Techniques for Deep Learning”. In: Pattern Recognition 137, p. 109347. issn: 0031-3203. doi: 10.1016/j.patcog. 2023.109347.

Yadav, A. et al. (2023). “Hybrid Artificial Intelligence-Based Models for Prediction of Death Rate in India Due to COVID-19 Transmission”. In: International Journal of Reliable and Quality E-Healthcare (IJRQEH) 12.2, p. 15. doi: 10.4018/IJRQEH.320480.

Yan, K. and D. Zhang (2015). “Feature selection and analysis on correlated gas sensor data with recursive feature elimination”. In: Sensors and Actuators B: Chemical 212, pp. 353–363. doi: 10.1016/j.snb.2015.02.025.

Yang, C.-C. et al. (2003). “Application of decision tree technology for image classification using remote sensing data”. In: Agricultural Systems 76.3, pp. 1101–1117. doi: 10.1016/S0308-521X(02)00051-3.

Ye, Z. et al. (2020). “Tackling environmental challenges in pollution controls using artificial intelligence: A review”. In: Science of The Total Environment 699, p. 134279. issn: 0048-9697. doi: 10.1016/j.scitotenv.2019.134279.

Ying, Xue (2019). “An Overview of Overfitting and its Solutions”. In: Journal of Physics: Conference Series 1168.2, p. 022022. doi: 10.1088/1742-6596/1168/2/022022.

Zhu, J. et al. (2020). “Deep transfer learning artificial intelligence accurately stages COVID-19 lung disease severity on portable chest radiographs”. In: PLOS ONE. doi: 10.1371/journal.pone.0236621.

Zhu, L., H. Jin, R. Zheng, et al. (2014). “Effective naive Bayes nearest neighbor based image classification on GPU”. In: Journal of Supercomputing 68, pp. 820–848. doi: 10.1007/s11227-013-1068-7.

Zhu, Q. et al. (2020). “Improving Classification Performance of Softmax Loss Function Based on Scalable Batch-Normalization”. In: Applied Sciences 10.8, p. 2950. doi: 10.3390/app10082950.